MacTalk 人生元编程



池建强◎著

青苹果数据中心出品

目　录

MacTalk·人生元编程

作者：池建强
策划：青苹果数据中心
封面设计：赖仙明
版式设计：邱霜

湖南省青苹果数据中心有限公司
湖南 长沙 开福区青竹湖大道399号

互联网出版许可证：新出网证（湘）字013号
电子邮箱：GA@egreenapple.com
网址：www.egreenapple.com
湖南省青苹果数据中心有限公司为作者和相关机构提供数字出版服务。
本书电子版如有错讹，敬请指正，我们会及时更新版本。


MacTalk

卷首语

开篇：为何而写

在盛夏的某个周末，我在杭州度过了一段美好的时光。见到了很多老朋友，认识了很多新朋友，参加了阿里嘉年华的夜聊节目，领略了夜行西湖的美妙，此行非虚。

杭州的天气和北京截然不同，北京要么是热烈的阳光直射大地，干燥温暖，要么是大风裹挟着暴雨席卷京城，要么以漫天的雾霭夹杂着浓浓的PM2.5气息为你带来这才是北京的直观感受。杭州就温和多了，阳光似乎总被一层薄雾笼罩，温度很高，但空气中总夹杂着水汽，出去转一圈浑身湿漉漉回来的一定是北方土鳖，偶尔碰到一个PM2.5，它会告诉你，“我要旅行去北方，那里才是我的家”，想想让人颇为绝望。说是有台风，但台风也只是轻轻掠过杭州上空，带来了些许清凉，带走了一些红尘。

现在我即将离开这座美好的城市，坐在开向北方的列车上，颇为惆怅的写下今天的MacTalk的主题：为何而写？

在阿里嘉年华的『夜聊』节目中，有一个问题是“你为什么要写东西？”，换个文艺一点的说法就是，你就是个写代码的，不去好好做程序员这份有前途的职业，为什么要来写文章呢？谁让你写了，能挣钱吗，有人看吗？你爸爸妈妈知道吗？你的老板允许吗？……

每个嘉宾都谈了自己的看法，我整理一下自己的思路，分了以下这么几点，看能不能把这事说的清楚一点。

1、我觉得程序员这个群体是非常幸运的一群人，我们生在一个技术改变世界的时代，而我们可能正在做着能够改变世界的技术，这是何等的荣耀和机遇。想想Apple的终端设备、Google的眼镜和汽车、Amazon的电子阅读、阿里的电商和金融梦，难以想象10年以后这个世界会发生什么样的改变。更不用提那些还在车库里酝酿的各种奇思妙想，而这些改变，程序员都将参与其中。

虽然我们现在依然会遭遇到这样的烦恼：

“大哥听说你是搞电脑的？”

“我做了十年技术了”

“好的，能帮我装个XP吗？要番茄版的。”

这时候就可以把兜里准备好的西红柿扔出去了，如果不会扭到自己的老腰，再加个360°回旋踢就更加完美。

2、不要给自己设限。谁说程序员只能写代码了，谁说程序员都是宅男了？程序员里才华横溢的多了去了，无论是《黑客与画家》还是《乔布斯传》都描述了很多具备文艺气息的技术大师，他们要么作画、要么弹琴、要么写作、要么运动，同时还写的一手好代码。其实万事万物都是相通的，要么熊样要么鸟样，如果你能够把代码写的很好，那么为什么不去把自己的思想和设计通过文字表达出来呢？如果你能够把技术文章写好，慢慢就能写出人文类的文字，慢慢你就会发现自己已经站在科技与人文的十字路口了。最高境界就是，你站在哪，哪里就有一个刻着科技和人文路标的十字路口。

在MacTalk之前，我写博客，也为一些媒体写文章，最好的时候一个月会写2-3篇，每次约稿都要拖到最后才能完成，但是开始写MacTalk之后，到今天为止，近8个月的时间我写了180多篇有效文章，平均每周5篇左右，文字总量在15万到20万，这个惊人的数量是我从来没想到的，而且花费的时间是我剪切了刷微博、浏览网页、扯淡、看电视的时间，粘贴到MacTalk上的，每个人时间就那么多，我觉得我并没有损失什么。

所以，不要为自己设限！

3、写作即思考。写作其实就代表思考，你需要言之有物，需要架构需要梳理，要有开端有结尾有结论有主题，特别神奇的是你构思了一篇文章，写完后发现文章像具备了生命一样生长出了很多奇异的果实，它们就在那些文字中间微微颤动，闪烁着独特的光泽，仿佛被岁月冲刷过的鹅卵石一样，而这一切你可能完全没有想到过，而且不可复制。所以有时候我们去看之前写的文章，会产生两种感觉，要么是觉得写的太烂了，怎么会写的如此臭不可闻；要么是觉得写的太好了，妈妈我再也写不出这么牛逼的文字了，我觉得这样两种感觉，都挺好。

优秀的写作者不仅能让事情变的容易理解，而且能够换位思考，沟通顺畅思维敏捷。与这样的程序员交流是赏心悦目的。遇到问题时他会抽丝剥茧，告诉你问题的前因后果，由表及里，并且把能够反映问题的各种信息等都提供给你，包括他自己尝试解决问题的措施和结果。

所以，为自己写作！

4、附加值。写MacTalk的初衷是给大家介绍一些Mac相关的技术和技巧，从来没想过商业化和媒体属性，但是做到现在，积累了几万读者，每天都有用户的反馈和赞助，你们告诉我这些文字为你们带来了欢乐、思考和启发，解决了问题，我告诉你们开心就好。MacTalk还为我

带来了很多好朋友，有知名大公司的，也有不知名的小公司的，还有在创业的。天地悠悠过客匆匆，能认识这些朋友，真好！

所以，为读者写作，附加值会随之而来。

MacTalk，为自己而写，为读者而来。但行好事，莫问前程；河狭水急，人急计生。



池建强，70后程序员，Blogger，微信平台MacTalk作者。先后任职洪恩软件和用友集团，从事互联网和企业应用软件研发，目前担任瑞友科技IT应用研究院技术负责人。

热爱技术和编码工作，Apple和Google产品重度用户，分享技术，坚持梦想。

博客：http://macshuo.com
微博：@池建强
微信平台：MacTalk By池建强（sagacity-mac）

MacTips快速查询

MacTalk



序

技术写作与减熵

池建强兄告知我他的电子书即将出版，作为他的微信公众帐号“MacTalk”的忠实读者，必须要支持一下。“MacTalk”和我的“小道消息”差不多同时期开始进行写作，我们也经常交流写作和运营的经验，从内容流行性上来说，我的“小道消息”赢得了更多读者和一点名声，但从内容上看，“MacTalk”则对读者更有价值，所以，一有机会，我总要推荐一下他的帐号。

这本书的几个主题中，我最为喜欢科技与人文的这一系列文章，阅读的过程中我也用自己的经历来印证，回顾自己从业过程中那些错误的抉择，给了我不小的启发，建议对自己职业生涯有困惑的朋友能读一下这个主题。如果是五年前看到这本书，我可能会更喜欢程序员与编程这个主题。而作为Mac用户，一系列的Mac技巧则直接提升了我的工作效率，这篇文章就是在MacTalk推荐过的工具上写出来的。

或许有人觉得，写书会带来经济收入和名气，但实际上，我个人认为这可能是一种误解，在我们这个领域，写书的确会带来一点“好处”，但肯定比很多人预期的要小得多，对于作者来说，投入和产出简直不成比例。如若不信，也可以先算一笔经济账，假定一本讲技术人文的书能卖到五千册到一万册，按照8％的版税来计算，也就是几万块钱的收入，这还没缴税呢，当然如果是出版电子版图书的话需要另作计算，但经济上的收入也不会有多高。而一个作者在这上面投入的时间和精力如果用来做点别的事情的话，比如出去做几次培训，或者是做个技术顾问收点咨询费什么的，轻轻松松也能拿到这个收入。至于“赚”名气，写再多的书怕也不如多参加几次行业会议多做几次公开演讲赚取名气来的快。

王小波称自己的写作是一个“减熵”的过程，通俗一点说就是“吃力不讨好”，考虑到王小波曾经也是个不错的程序员，套个近乎，算是咱们这个行当的，所以这一点上IT人倒是容易对他的文章有共鸣，至少我就受到了一些影响。我曾在微博上说过，尽管曾被怂恿或是鼓励去写本书，但是总还下不了这个决心。一方面是考虑到经济账，另一方面，心里也有点担心总干这个“减熵”的事情是不是有点自己跟自己过不去？还有一个不好意思说的原因是我一想到像建强这样把文章重新整理一遍就

头大，总是有畏难情绪。但是世界上总是有人在不停的做这种“减熵”的事情，我想这么做的人可能就是因为这是一种乐趣，一种真正的乐趣，尤其是你的文字能够帮到人能够启发别人的时候，那种快乐难以言说。

作为读者，我能做到的事情并不多，想来想去，唯有尽可能的支持作者的创作，并且向更多人推荐这本书。顺便说一下，如果他这本书销量好的话，不排除我也会“减熵”一次。

冯大辉[注]

注　释

[注]丁香园技术负责人，网络ID：Fenng
微信公众帐号“小道消息”作者

但行好事，去他妈的前程

MacTalk微信推送的文章我都读过，结册成书后我又买下读了一遍。看过整册后，我竟然产生了一种“敬畏感”。

这种敬畏感源自我心中对“写了十几年代码的程序员”这样头衔的赞叹和崇拜——每一个写了十几年代码的程序员都是一本书。

我也差点儿能成为一个“老程序员”，我父亲该算是国内第一代程序员，我出生时他在一所大学教编程，而我母亲当时在另一所大学教统计。诞生在这样的一个家庭，我几乎在育婴室就被贴上了“很可能会是个码农，请注意安全”的标签，接生的护士说这小宝贝好可爱，来给姐姐笑一个，笑，笑，你特么倒是笑啊，我冷漠而淡淡地回应，bad command or file name。

我曾有过在短时间内带着极大的心理压力，以极大的工作量写代码的经历，我甚至不太敢回忆那时的感受。我知道那种长时间面对一台机器的冰冷和绝望。机器没有感情，它执行你要求的每一句逻辑，你对它没什么脾气，因为起承转合都来自于你的编排，机器只是精确执行。

有人评价程序员说他们“成天跟机器打交道”，别闹了，电脑才不会跟人打交道，在这台坚硬的机器前，一切喜怒哀惧都是程序员心中的自言自语。

于是很多人觉得程序员很可怜，木讷、神经质、不善交际、不修边幅，嘲笑程序员的段子足够装一火车。我听到不少姑娘偷偷地跟自己的闺密说“别嫁程序员，没情趣”，或者“嫁个程序员，老实有钱死的早”。

在这种诡异的环境中，很多程序员都在“转型”和“突破”，做产品，做运营，做职业吹货，他们想办法甩开“做技术的”或“写代码的”标签。我不止一次听到“程序员是吃青春饭，你还是得赶紧转型”这样的所谓前辈忠告。

如此背景之下，“一个写了十年代码的老程序员”头衔的背后，似乎一定有一曲二胡拉就的挽歌，你忍不住掏出纸巾想递给对面已经年逾不惑的他，伸手叫服务员“再上一箱啤酒，冰的”，你悲怆，你泣涕涟涟，你起好了范儿，准备听他开讲。

结果他一开口，竟然是……

活泼幽默轻松直率荤素搭配清爽可口。

“可是船长，这明明是一场悲剧，你笑什么呢？”

“谁特么说这是一场悲剧了”

这本书就是这样一个老程序员哼的这样一段小曲儿。端起来有很多技术性文章，放下去有嬉笑怒骂和语重心长。

当然了书名叫做MacTalk，自然有不少Mac技巧。看得出他对苹果情有独钟，书中开篇就是Mac的历史和故事。他还有一篇被广泛转载的“趣谈个人建站”，细致到代码级别。他还写人，写书评，说是书评，实则是自省、梳理和复盘，他读《黑客与画家》，势必跟我这个只写过几行代码的产品狗读起来深度不同，我喜欢他那篇书评。

他有代码情结，感觉就是随便一撩动，他就马上坐那儿不起来非要给你写上一段儿，还拉着你让你看他写，你瞅瞅书中字里行间那些代码，像着了魔似的。

书里我最喜欢的一篇文章是《人生元编程》，他做了一个跨度非常大的类比，用元编程的思想类比人生，那段时间我正在看《自私的基因》，我在这三个概念中间也找到了些似是而非的暗合。

我觉得这篇东西是程序员用程序机制思考人生的一个缩影（当然了书里还有些其他的缩影比如并发什么的），如果你只是一个把堆代码当差事的程序员，我特别推荐你看看这一系列文章，它们或许会帮你打开一扇门，看看这种奇异的关联。

比如你可以在爱情中拒绝GC，自己照顾（take care）你的每一段回忆（memory），多浪漫啊。

他认同对他文笔“相对轻松温和”的评价，他说自己在网际多年看惯了刀锋和鲜血，所以他不愿意参与或挑起争端。但我不同意他的文笔“温和”，我觉得不温和，而且挺有情绪的。有情绪的文章读起来像作者在跟你聊天儿，也很容易有代入感，是好事儿。

最后想提一下这本书的价格，这本书现在卖2块9毛9，之前经常有人跟我抱怨推荐的书太贵，买不起。这本……应该，还好吧。

这本书有52篇文章（其中还有130个MacTips），平均每篇文章5分多钱，真是卖得够贱的，要说作者好歹也是正儿八经一个帝都仔，到底图了个啥？他在序言里是这样回答这个问题的——“但行好事，莫问前程”，我特么太喜欢这句话了！

我想把它送给每一个在深夜烧烤摊前黯然神伤的人们，精于算计的人们，小心翼翼的人们，举轻若重的人们，当然也包括我自己——但行好事，去特么的前程。

泰山崩于前，我依然沐浴更衣焚香沏茶，诚心正意，手起键落

Hello World!

邱岳

微信公众帐号“二爷鉴书”作者

生命中遇见的每一本书，都不是偶然

霍夫斯塔特（《哥德尔，埃舍尔，巴赫，集异璧之大成》的作者）在给内格尔的《哥德尔证明》一书作序时，写过这样一段令人印象深刻的话："……当时是1959年秋天，在门罗公园的开普勒书店里，我完全偶然地见到了《哥德尔证明》。可能此刻在中国，有个人——可能就是你——正在书店里面随意浏览这本书，正在翻动它的书页并正在读着眼前这些词句。或许如果你买了这本书，会使你的生活发生革命性的变化，就像这本书对我那样！当然，也可能什么也没有发生。也可能你根本就不在书店里。或许你还正在睡觉呢！但是不管是哪种情况，不管是你或是别的什么人，我的确希望在中国有人能发现这本书，能感受到它是如此之美妙，如此激动人心……"

按理说，作为一个没有Mac，不怎么懂编程的人，我和MacTalk应该是没有任何交集的。不过现在仍然记得第一次看到MacTalk的那一天：一个初夏的中午，阳光很好，看到Fenng在小道消息里推荐了Mac君的公众账号，正好吃完饭闲来无事，于是顺手加上了，然后就看到了沉没成本那篇文章，正好之前在曼昆的经济学原理里看到过这个名词，印象很深刻，于是查看历史消息，发现作者在精通技术之余，还有着深深的人文情怀，这在码农界是不多见的，于是慢慢变成了Mac君的忠实粉丝，每一篇文章都能给人启发，更难能可贵的是，作者对每一条回复到公众账号的消息都会认真阅读并且尽力解答读者的问题，这在很多傲娇的公众账号中更是不多见的。

然后Mac君就出了书，在多看买了，又看到豆瓣也上架了，说实话，对于我这种早已告别了文艺青年这个称号的人来说，实在不好意思来这么小清新的豆瓣发评论，不过，这本书确实给了我很多启发，所以，写下这些，也是为了让更多人能够有缘和它相遇。

乔帮主在著名的"遗失的访谈"里曾经说过：每个人都应该学习一门编程语言。我个人认为，这句话是非常有道理的，不仅仅是为了会写代码，能够实现自己的想法，更重要的，是通过这种训练，能够培养程序员的思维方式。正在我为学习哪门语言难以选择时，Mac君适时推出《如何学习一门编程语言》这篇文章，以一位过来人的身份提点后学。在他的推荐下，开始学习python（和那位码模木有关系，完全是因为今年是蛇年~），有人说，养成一个好习惯需要21天，坚持到现在，每天写几段代码已经成为我的习惯，也许在高级码农眼里，我只是一个非常

非常初级的入门者，但是对我自己而言，能够从coding中获得乐趣，将来某天有能力实现自己想完成的事情，就已经很令人满足了。

我一直认为，书和人是有特殊的缘分的，缘分这个词也许有些矫情，更确切的说，遇到的每一本书，冥冥中与你都有特殊的connection。在之前的印象中，码农都是一群带着高度近视镜，脖子僵硬，弯腰驼背的nerd（s），对人生有着令人不敢苟同的奇怪看法。但是Mac君这本书的出现，很大程度上颠覆了这种印象，作者不仅具有high level的专业技能（在百余篇MacTips里有充分体现），而且行文生动，文字幽默，在保证知识性之余，也兼顾了全书的可读性，再加上这么便宜的定价（2.99元的价格估计现在连一枚可爱多都买不到），不买本支持一下，怎能对得起这难得的偶遇呢？

正如题目《人生元编程》所暗示的，这本书也是作者思考与自省的结晶。所谓“元”（英文“meta”），就是指能够对自身状态进行描述。例如希尔伯特当年所说的“元数学”，就是指关于整个数学系统的语言，那么元编程，就是能够操作代码的代码，人生元编程，就是具有自省能力，随时检查和控制自身情绪和行为、思考自己想法、改变大脑的动机……或者，用一句简单的话来说，就是对当下的状态保持清醒的“觉知”。

继续用霍夫斯塔特的话来结尾吧：“……如果你问我是否取得了最后的成功，答案是‘当然没有！’如果是的话，生活将会变得令人厌烦。如果人的心灵会被化简为几条僵化的规则，即便是相当大的一个僵化规则的集合，那会是一件令人极度悲哀的憾事……我们是幸运的，因为我们的心灵是如此不可预知；正因为如此，生活才充满了情趣。尽管如此，我们仍在进行努力来科学地了解我们自身……”

希望每位有缘读到这本《人生元编程》的读者，都能像作者期待的那样，具有人生元编程的能力，洞察自身的微妙与精深。

作者：密码有误


MacTalk

Macintosh的命名

【发布日期　2013年6月27日】

1976年，美国有两个叫Steve的孩子开了一家以水果命名的电脑公司，叫做苹果电脑公司，为什么叫苹果，据说是那阵儿乔布斯天天吃素，顿顿水果餐，瘦的跟干煸豆角一样，他闪烁着迷离的眼神（估计是饿的）和沃兹说，“我刚从苹果农场回来，咱公司就叫苹果电脑吧，阳光健康，还不吓人，有科技有人文，在电话黄页上还能排在前列，多好”，于是苹果电脑公司就诞生了。

现在我们知道，苹果公司所有电脑产品的命名都和Mac相关，比如MacBook Pro，MacBook air，iMac，Mac Pro等等，但是第一代苹果电脑却和Mac毛关系没有，人家叫Apple。开天辟地的是Apple I，居功至伟的是Apple II，以前写沃兹小传的时候说过，这两个神作都是沃兹单枪匹马做出来的，乔布斯也就是焊焊板子买点披萨神马的。

有了Apple I和Apple II垫底，很快就衍生出一些新产品项目，我们把时针拨到1979年，那时苹果公司有四款产品在研发，AppleII，AppleIII，Lisa和本文的主角——Macintosh。其中AppleII一直充满活力，在退出历史舞台之前都是苹果公司的支柱产品，AppleIII和Lisa就比较惨了，命运多舛，一会万般宠爱，一会无人问津，这种境遇就不可能做出好产品，结果AppleIII只生产了9万台，Lisa更可怜，1983年推出，1986年彻底终止，余货被埋在犹他州的垃圾堆里。这时候Mac的原型正在孕育，这是个微不足道的小项目，项目名称叫做“安妮”，项目的负责人是杰夫·拉斯金。



大家一看到“安妮”二字是不是立刻就懵了，洋气的苹果公司居然有这么基情四射的项目名称，但是具体为什么叫安妮，我也无从考证，大家还是来认识一下项目经理吧，这兄弟叫做杰夫·拉斯金，同样是一位技术牛人，是苹果的第31位员工。苹果公司的开创者们似乎都脱不开人文与技术的情怀，拉斯金的专业是计算机科学，但是教过音乐和视觉艺术，在厌倦教书之后，就租了一只热气球，跑到校长家上空大声喊道，我特么不干了。谁这么辞过职？

拉斯金在自己的博士论文里写着，计算机应该是给人用的，不仅仅是给极客黑客各种客用的，除了神秘的命令行，还得有图形界面。所以他对世界宣布：“I have a dream，那就是为大众制造价廉物美的电脑”！

想到做到，1979年，拉斯金说服了当时苹果公司的管理者迈克·马库拉，成立了一个小规模的项目组用来研发廉价的、同时具备图形界面和命令行的电脑，当时这个项目代号就非常神奇的被命名为“安妮”，咱也不知道拉兄是怎么想的，终于有一天，拉斯金觉得这名字太女性化了，于是主动更换了项目代号，用自己喜欢的一种苹果（又是苹果！）来命名，叫做麦金托什（McIntosh）。但是为了避免与另一家音响设备制造商麦金托什实验室（McIntosh laboratory）重名，拉斯金对这个单词做了些微调，改为大家现在熟知的Macintosh。

感谢拉斯金，如果不是他把Mac的名字改了，现在大家用的都是iAnnie和Anniebook，那感觉得有多么怪异。

1984，Mac诞生

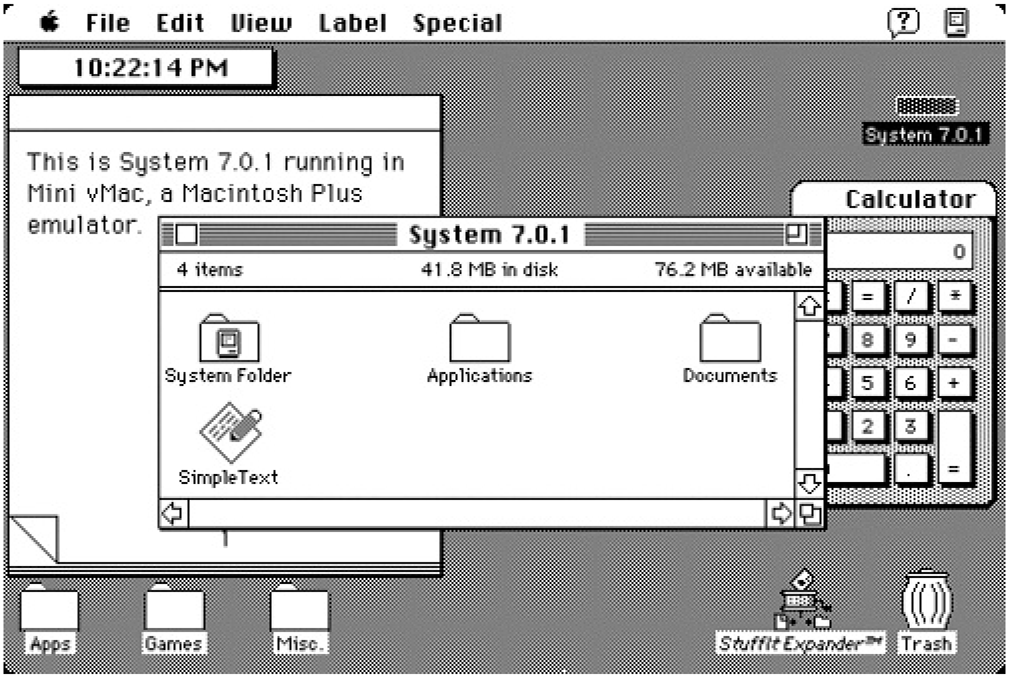
【发布日期　2013年6月30日】

“所有伟大的事业都源于梦想，始于微不足道”。

拉斯金的梦想是“数以百万计的电脑”，是“如果个人电脑能够真正面向个人，那么任何一个家庭都该拥有一台”。但是Macintosh项目初期，整个项目组只有四名研发工程师，四名啊，各位兄弟，这是在做操作系统和硬件工业设计，不是在写移动App！就像秦始皇告诉蒙恬，“兄弟，哥有件事托你办一下，你挑选20名精锐勇士，去修一下万里长城，Okey？You can do it！”，但事实就是这么残酷，搁到现在您就算打破脑袋也找不到这样的团队！

这还没完，人少不算什么，您别没事就来捣乱啊。每隔一段时间Mac项目就会被拿出来讨论一下，讨论的主题不是加人加钱，而是啥时候解散？为神马还不解散？好在拉斯金也不是吃素的，每次都能凭借三寸不烂之舌让项目化险为夷，又惊又喜。终于有一天，Macintosh被乔布斯盯上了，这下拉斯金老实了，“他妈妈再也不担心他的项目被解散了”，而是什么时候这个项目会被小乔据为己有。

根据乔帮主的各种言行访谈，我相信他对编程有着深刻的认知，但是根据各种史料传记，帮主应该是没有真正写过代码的。没编程但是对编程有深刻的认识，这种行为是无法解释的，所以史称“天才行为”！



虽然不会编程，但是乔布斯非常欣赏拉斯金的想象力，在Lisa项目折戟沉沙之后，乔布斯盯上了Macintosh，不过在帮主的词典中是没有“价廉物美”这个词的。他告诉拉斯金，“你太不高端大气了，搞什么价廉物美呢？咱有钱，不用考虑电脑的成本，要做就做完美的产品，可操作性、图形界面、性能，一样都不能少”。洗脑完毕，乔布斯开始让工程师为Mac换上性能强劲的摩托罗拉68000，用来支持鼠标和华丽的图形效果。工程师这一次站到了乔布斯这一边，哪个程序员会不希望自己的程序跑在更好的硬件上呢？

终于，乔布斯代替拉斯金接管了Macintosh项目组。当时的乔布斯迫切希望用Macintosh再次证明自己，他开始动用自己的资源和现实扭曲力场来改造和影响Macintosh项目组成员，同时也吸引了更多优秀的技术天才加入Macintosh的研发。乔布斯甚至不切实际的希望Macintosh能在1982年左右发布，他的激情和工作方式虽然让项目组成员间歇性崩溃，但同样在激励着他们奋力前行，实现了很多似乎不可能完成的天才技术创意。

Macintosh研发期间那个著名的缩短Mac启动时间的故事尤其让人印象深刻。有一天乔布斯走进负责Macintosh操作系统的工程师的办公隔间查看电脑演示，随即开始抱怨系统启动时间太长了。像所有程序员一样，那个工程师开始解释balabala……，乔布斯打断了他，说，如果能够救人一命的话，你愿意让系统的启动时间加快10秒吗？工程师说可以考虑。乔布斯拉过白板开始计算，如果500万人使用Macintosh，每天开

机多用10秒钟，那加起来每年要浪费的时间大约是3亿小时，如果我们把这些时间节省下来，那每年就相当于挽救了数以百计的生命。

工程师一听吓尿了，几个星期后乔布斯回来一看，系统的启动时间减少了28秒。年轻时的乔布斯虽然工作方式方法上有很多值得探讨的地方，但是他能够看到宏观层面的东西，从而激励别人奋勇向前。

终于，时间来到1984年，在经历了数不清的延期之后，Macintosh要发布了，乔布斯为此策划了著名的“1984”广告，这是一个注定名垂千古的广告片，但第一次给苹果董事会成员播放时，大多数人居然认为这是看过最差的广告片（你们就知道大部分光管理不干活的人有多么没品了吧）。不过乔布斯和沃兹都非常喜欢这个广告的创意，甚至沃兹提议广告费由他和乔布斯一人一半承担（厚道的沃兹）。经过努力，1984广告按照原计划播放。

1984年1月22日，在第十八届美国职业橄榄球“超级碗”大赛中，苹果公司播放了这段构思独特的广告片“1984”。广告借用乔治·奥威尔的小说《1984》映射IBM对市场的垄断和不思进取，把Macintosh作为挑战IBM的新生力量，在广告的结尾，屏幕上出现广告词：“1月24日，苹果电脑公司将推出Macintosh电脑，你就明白，为什么今天的1984不会变成《1984》”

“30年过去了，IBM似乎又开始不思进取了，但这次IBM的角色已经从当年的老大哥变为时代的follower，能否跟上还要看其自身的变革力度”

这则广告取得了令人震撼的效果，最终被《电视指南》和《广告时代》评为有史以来最伟大的商业广告。

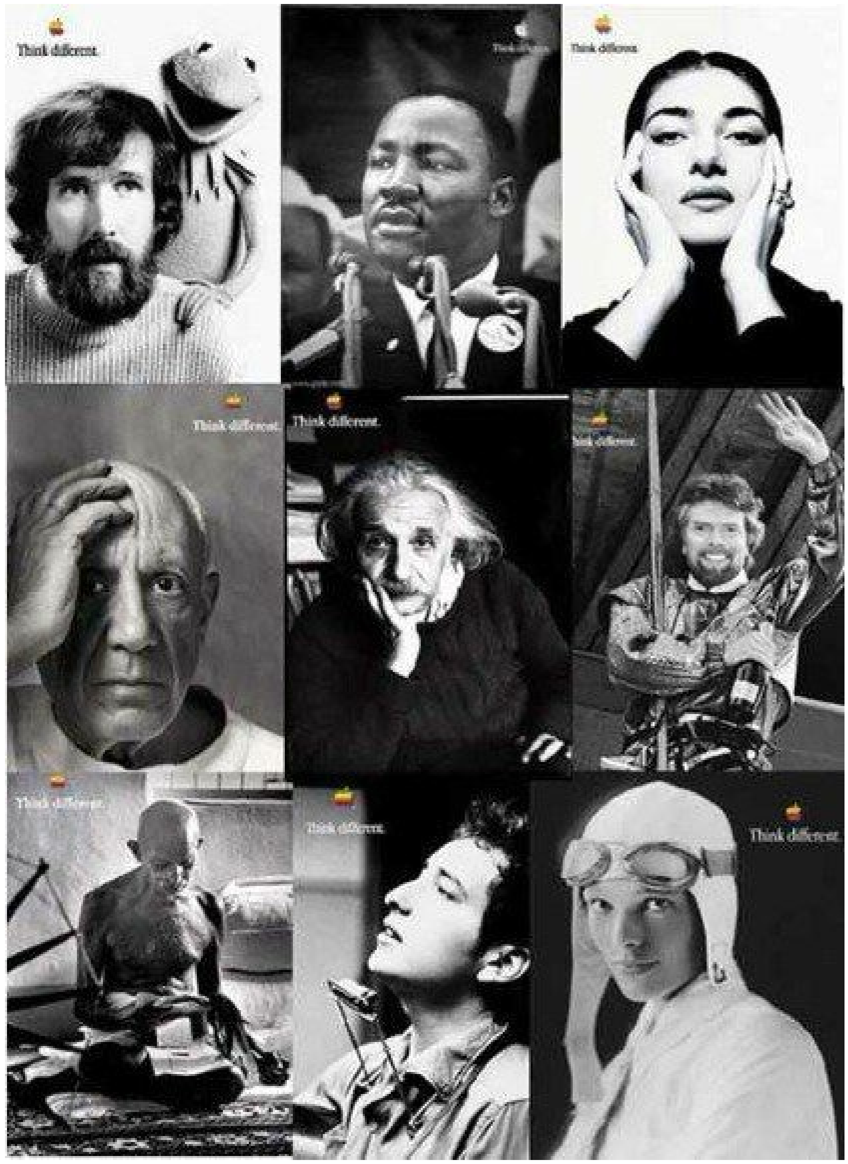
从此，Macintosh拉开了另一个伟大时代的序幕。

非同凡想

【发布日期　2013年8月6日】

6月底我在“1984，Mac诞生”中介绍了苹果公司的一则著名广告“1984”，在那篇文章中我描述道：

1984年1月22日，在第十八届美国职业橄榄球超级碗大赛中，苹果公司播放了这段构思独特的广告片“1984”。广告借用乔治·奥威尔的小说《1984》影射IBM对市场的垄断和不思进取，把Macintosh作为挑战IBM的新生力量，在广告的结尾，屏幕上出现广告词：“1月24日，苹果电脑公司将推出Macintosh电脑，你就明白，为什么今天的1984不会变成《1984》”



不过我最喜欢的苹果广告并不是“1984”，而是今天要介绍的“非同凡想——Think Different”。2011年乔布斯辞世的时候，这条广告曾经被广泛的报道过，喧嚣过后，我们再来品味这些充满时间感的黑白画面和优美到极致的文字，想想宇宙苍茫，想想在时间的滚滚洪流中推动人类进步的人、事、欢乐、苦难和疯狂，我们就知道，改变世界不是闹着玩的。

这则广告诞生在乔布斯刚刚回归苹果公司的那段时期（1997年），乔布斯还仅仅是头顶“顾问”头衔的领导者，连临时CEO都不是。苹果当时内忧外患，除了要精兵简政，缩减产品线，拓展优势产品之外，还要告诉外界，苹果依然生机勃勃和与众不同，这个重任就落到了“非同凡想”上。

苹果找到了当时知名的广告公司TBWA/Chiat/Day，这个公司也是“1984”的缔造者，大家开始坐在一起寻找创意。与“1984”一样，创意必定伴随痛苦，整个过程充满了争论、反击、争吵、骂娘和痛哭流涕，双方恨不得杀死对方，剧情狗血得能写一个剧本。参与创意的人逐字逐句的推敲文字、画面，选择历史人物，选择旁白者等等，终于，在所有人彻底崩溃之前，广告内容完成了。

在旁白者的选择上，有人推荐乔布斯，但是在当年看来这并不是个好主意，乔布斯那时的成就很难比肩画面中的其他人物。最后大家选择了奥斯卡历史上最年轻的影帝理查德·德莱弗斯，不过，乔布斯也录制了一个版本，所以我们今天才有幸能够听到乔布斯本人旁白的“非同凡想”。

在广告创意和制作接近尾声的时候，乔布斯决定正式接管苹果公司，由顾问变为iCEO（临时CEO）。“非同凡想”广告播出后，所有人都知道苹果又活了，他们重新开启了复兴之旅。

16年过去了，让我们再次欣赏一下这则广告的文字版和乔布斯配音的广告片：

在此向那些疯狂的家伙致敬
他们我行我素，桀骜不驯，惹是生非，就像方孔中的圆钉一样
他们用不同的方式看待世界
他们既不墨守成规，也不安于现状
你尽可以支持他们，反对他们，赞美或诋毁他们
但是唯独不能忽视他们
因为他们改变了世界
他们让人类向前跨越了一大步
他们是别人眼里的疯子
却是我们眼中的天才
因为
只有疯狂到自以为能够改变世界的人
才能真正的改变世界

影片中包含的人物有：阿尔伯特·爱因斯坦、鲍勃·迪伦、马丁·路德·金、理察·布兰森、约翰·蓝侬、巴克敏斯特·富勒（Richard Buckminster Fuller）、汤玛斯·爱迪生、穆罕默德·阿里（Muhammad Ali-Haj）、泰德·特纳、玛丽亚·卡拉丝、圣雄甘地、阿梅莉亚·埃尔哈特、亚弗列·希治阁、玛莎·葛兰姆、吉姆·韩森、弗兰克·劳埃德·赖特、毕加索。广告的结尾，一位青春洋溢的小女孩，睁开了紧闭的双眼，仿佛看到无限可能……

怀念2007

【发布日期　2013年3月21日】

看到一则消息，或叫传闻，苹果将在6月29日，也就是iPhone发布6周年的时候发布新产品。届时苹果将举办一个名为“Original Passion, New Ideas”的活动来庆祝iPhone发布6周年，并在活动中发布下一代iPhone和iPad等产品。由于苹果传统的WWDC大会也是在每年6月举行，不知道这两个活动是不是一起进行。

作为技术人员，相对于新产品发布会，我更喜欢每年一度的WWDC大会，届时会有年轻的、年迈的、大师级的各种产品经理、布道师、程序员来讲解他们在设计、开发、创意中的想法和实践，几十个session，看起来非常过瘾。今天的附图就是2012年的WWDC Logo。

时间如白驹过隙，距离第一部iPhone发布已经6年了，我手机上有一个用来做纪念日提醒的App显示，第一部iPhone发布于2007年6月29日，至今2092天。6年时光，智能手机和平板领域发生了翻天覆地的变化，当年那个手持iPhone睥睨世界乔布斯已经仙去，Android崛起，与iOS分庭抗礼。手持设备上超过百万的App数量彻底改变了人们的生活，一切从iPhone开始。



iPhone的各个版本中，第一代和第四代是获得赞誉最多的，第一代横空出世，告诉世界现在和未来，手机应该是这样的。三年以后第四代iPhone破茧而出，工业设计和软件技术的水准堪称空前，远远的把同年代的手机落在身后70码，颇有倚天不出谁与争锋的气势。

时至今日，无论iPhone或iPad有多么大的改进，比如芯片、摄像头、NFC、硬件工艺和外观等等，都不能再给我们带来创世的感觉。除非出现颠覆性的特性，比如电池续航能力的突破，全息影像，智能感知等等。毫无疑问，苹果会把iPhone、iPad、Mac和App生态帝国做得越来越好，我们可以慢慢等待，也可以去添砖加瓦，但是我们更期待那种颠覆性的产品出现，就像Mac、iPod、iPhone、App Store、iPad，这才是苹果创新的基因。未来的颠覆是什么呢？可能是iWatch、iTv，或者是我们没有想到的东西……

作为一个软件从业人员，我知道一个产品从无到有是困难的，从有到精是艰难的，而当你站上一个巅峰之后，哪怕是做最微小的改进和提升，都需要花费大量的人力物力，同时还要承受失败的风险。我们都知道，从平庸到优秀是容易的，从优秀到卓越是痛苦的，可能很多人、公司穷尽一生都无法达到卓越的境地。

有时候，我们只是需要一点耐心，把我们自己的事情做好的同时，再等等。

年轻时的梦想还在吗？

【发布日期　2013年3月17日】

每个孩子从小都怀揣梦想，积极上进，在充满无穷可能的选择里为自己编织最灿烂的明天。我想大家小时候都这样吧，没人从小立志要做个坏人或者流氓，很多时候别人问你的理想是什么啊？回答多半是科学家、作家、医生、老师，从来没见过一个孩子从小信誓旦旦的要做一个黑社会或贪污犯。即使人们长大了，由于各种原因变成了坏蛋和流氓，他们也希望自己的子女将来做个好人而不是坏蛋。



但是，你年轻时候的梦想只是你生命中的一个Logo，而且是个会变的Logo，比如苹果最初的Logo是这样，后来变成了这样（见附图）。我小时候因为作文被老师读，想当作家，因为参加数学竞赛，想当数学家，因为练毛笔字，想当书法家……这些都不重要，重要的是你在长大的过程中没有把这个Logo丢掉，你还在想要做点什么或改变点什么，你没有变成小时侯痛恨的那些人，这就够了。

人长大总是伴随着痛苦，现实变得越来越现实，那些无限可能都变成了没有可能或很小可能，这东西是很残酷的。当年我进入洪恩软件的时候，大概24岁，洪恩的总裁也姓池，比我大4左右，清华的本科生，当时已经拥有一家几百人的全国知名的科技教育公司。那时我暗暗的想，自己在28岁的时候，也要发展成什么样......这就是我当时的梦想。于是拼命工作、彻夜编程、努力学习，但是毕竟资质和眼界在那摆着，28岁了，有提升，但是远远到不到我设定的梦想。这时候你就会慢慢认识到，想取得什么样的成就，或者想成为什么样的人，除了通过长期艰苦的练习和有意识的提升，资质、环境、勇气、运气等同样重要。

然后很快你就三十了，三十而立，梦想似乎已经远走，每个人都在感慨，这么多年付出了真心却收获了一堆下水，不知道那些真心和梦想都去哪了，不是说能量守恒吗？都他妈去哪了？其实都在，它们就藏在某个角落默默的等着，一旦你准备重新上路，它们就会跳出来，指引你

前行。所以最重要的是你还想做事，而不是混世。有人说不知道自己想做什么，那么最好的办法就是把目前正在做的事情做好，如果你把不感兴趣的事情都做好了，一旦你找到自己的方向，那你得做得多好啊。

我自己离三十已经很久远了，正在加速冲向四十不惑的那条金线。子曾经曰过，四十而不惑，好像是告诉你到了40岁就什么都明白了，我曾经期望着大彻大悟的那一天的到来。后来我发现子就是个骗子，他的意思是你到了40岁，该明白的都明白了，不明白的估计你也不想去明白了，所以就不惑了。但现实的情况是，如果你一直在思考、读书、实践，做自己认为对的事情，到了40岁，你会发现不是不惑，而是有了更多的惑。这种未知的恐惧可能会伴随我们的一生.....

附图是苹果Logo的变迁，大部分人熟悉被吃了一口的苹果图案，但不知道第一个图是什么意思。那是苹果公司最早的合伙人韦恩设计的第一代苹果Logo，图案是一个缠绕了缎带的徽章，徽章正中是牛顿在苹果树下读书的场景。所以说乔布斯的苹果和牛顿的苹果，还是有点关系的。

乔布斯、沃兹和韦恩创建了苹果公司，但韦恩在公司成立没几天就卖掉了公司股份退出了，因为他觉得乔布斯和沃兹太孩子气了。多年以后苹果成了伟大的公司，有人找到韦恩，问他是否后悔退出苹果，他的回答是，一点也不，我喜欢现在的生活。所以说命中有时终须有，命中无时莫强求，后悔也没用，还不如不后悔！

品评**OS X Mavericks——**唯快不破（上）

【发布日期　2013年8月12日】

2013年6月11日，苹果的WWDC大会结束的第二天，我就升级了iOS7，还写了两篇文章。那时就有读者问，为什么不顺手把Mavericks也升了呢？



其实无论是iOS的beta版，还是OS X的开发者预览版，都是为了开发者测试和适配现有软件程序并提交bug的，苹果不会保证其可用性，尤其是前几个版本，系统崩溃、变砖、发热耗电、断网都非常可能，你看着千百个App在你面前挥舞但一个也不能用时是不会有人同情你的，甚至可能有代码界的达人慢慢的踱过来安慰你说，“该，谁让你丫升级了！”，让你觉得世界残酷无比。那我为什么升级iOS 7呢？第一，我觉得出了问题自己能Hold住，第二，别看大家每天走路开车冒着被撞飞和撞飞别人的风险低头看手机，但是手机只要能够打电话和发短信，就不至于耽误大事。OS X就不同了，那是工作和学习的必须品，我可不愿意冒风险花时间去折腾Mac，所以Mavericks一直没升。

前几天Mavericks的第五个预览版发布了，先问了几个朋友都说比较稳定，后外事不决问Google，得知我必须用的几个软件都已经有新版本兼容了，遂决定周末花时间升级，并在微博上发布了这一消息。这时

就看出人品了，马上有达人发来祝福，“无数事实证明了一个颠扑不破的真理——不作死就不会死啊，兄弟”，我毅然回复，“哥行走江湖靠的是个稳字，早就备份裊”。

插播一句，自从开通了MacTalk，有无数的人跑来问文件误删除了怎么办，系统挂了怎么办，硬盘坏了怎么办？一问之下都是没备份过的，为什么不备份，一部分不知道，一部分嫌麻烦。在这里我只能通过MacTalk再次告诉大家，“苹果为OS X开发了一个叫做时间机器（Time Machine）的软件，不是让大家混不下去的时候穿越到清朝去找格格玩的，而是为了备份你的操作系统和重要文件的，Mac君只能帮你们到这里了……”。

于是，我周末准备了茶、咖啡、啤酒和披萨，一边欣赏羽毛球世锦赛，一边完成了Mavericks的升级，整个过程无比顺利（人品好没办法），下面聊聊升级过程和使用体验。

Mavericks的命名

OS X向来是用我心爱的大猫命名的，比如Leopard，Lion等，这次不知道怎么发神经用了个加州冲浪胜地的名字，让我非常困惑，到现在也不知道中文名字是什么。而且这个名字似乎很难记，我发微博时把rick写成了risk居然没有被各类“错字控”和“错字党”发现，可见一斑。总之，大猫没了，让人伤感，而且，你不知道下一个版本会叫什么名字，难道找个攀岩胜地么？

安装

再说一次，安装之前先用Time Machine进行系统备份，保证你可以随时恢复到现在的系统状态。

下载OS X Mavericks 10.9的第五个开发者预览版，把安装程序拖到应用程序文件夹，双击安装即可。因为是升级安装，所以过程傻瓜的令人发指，选择一下安装磁盘，剩下的事情就交给安装程序了。我看完一场羽毛球比赛后回来，发现系统已经升级完成。

有童靴问什么是升级安装？这么理解吧，有一天你很累，早早就睡下了，这时候一个装修队趁着夜色的掩护偷偷进入你的豪宅进行了一次升级装修。等早晨的闹钟轻轻敲醒沉睡的心灵，你慢慢张开自己的眼睛，发现一切似乎都不一样了，但所有的东西都在，家具和电器换了新的，但衣物、摆设、位置都没变，甚至你上次打开的电视频道都保留下来了，总之系统已经不是那个系统，但你的个人设置（Profile）都保存下来了。OS X系统的每次升级都是如此，安装完成后几乎所有软件都不需要重新安装，甚至你打开文件的进度、Safari的Tab页都会被作为状态保存下来。我从第一次使用Leopard之后再没有重装过系统，要么升级要么恢复，就是这

样。

用户体验

升级完成登录系统，我的第一感受是“MD被骗了，什么变化都木有啊魂淡”，接着才进入第二感受，以下均为第二感受的体验。

1、去材质化

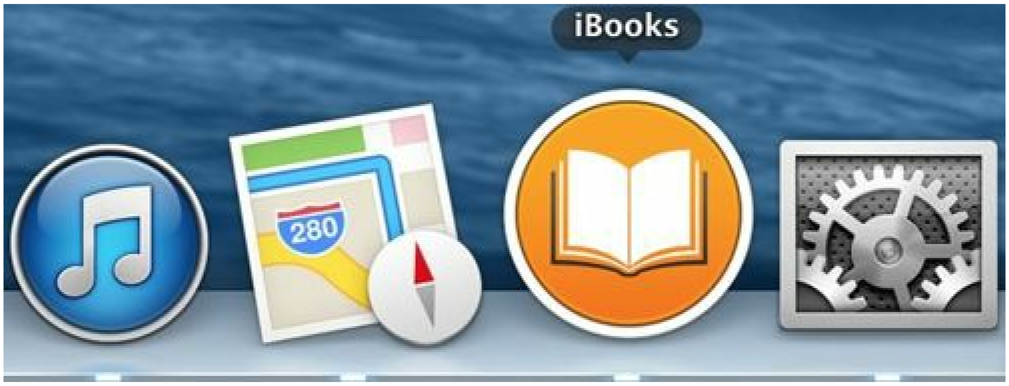
其实看到登录界面就会感觉有所不同，但由于只是背景材质的改变，所以容易让人忽视。Lion系列的登录界面背景是亚麻材质的灰黑色图片，苹果的Logo也带有镌刻的立体感，Mavericks的登录则极为简洁，纯灰色的背景和白色的Logo，登录用户头像也被妥妥的拍扁了。通知界面的背景也做了同样的改变，由亚麻背景改为纯色。Launchpad则彻底与当前桌面的色系融为一体，取消了二级文件夹打开显示纹理背景的设计，这一点与iOS7的毛玻璃效果有异曲同工之效。Dashboard空间的背景也由原来立体的圆变成了横竖交织的线。



至此，就我的观察，Mavericks基本完成了所有背景材质的去除，从早期充满贵族气息的金属拉丝和近期沉稳含蓄的纹理设计，都不会出现在新的OS X中了，一个时代已经结束。

2、扁平设计

Mavericks在OS X的扁平设计上做了一些尝试，比如登录界面，新应用iBook和地图的图标，日历、通讯录、备忘录和活动监视器的新设计，都显示了OS X的扁平化趋势，那为什么没有对所有的原生程序进行全新设计？我估计一个是时间上的问题，另一个是OS X的演进一直是潜移默化的，可能还不到大变脸的时候。老话，路要一步一步走，走的过大，容易扯着淡。另外，OS X用了这么多年，我确实还没有产生类似iOS那样的审美疲劳，那就让扁平来的缓慢一些吧。



3、细节改进

* 手势：四指划开显示桌面的时候可以通过手指的动作实时控制程序的位置，目前只有显示桌面和桌面空间切换可以做到类似效果。

* 万般委屈的Finder终于有了全屏功能，到现在我也想不通为毛原来就不让Finder全屏。另外，在Finder中选择文件列表时选择区域也变成大圆角了。

* 原来按一下电源键就会弹出是否关机的窗口，现在按一下关闭屏幕，长时间按弹出关机窗口，再长时间按强制关机。

* 点击顶边栏的电池状态，会显示使用大能耗的应用程序，如果你在用电池供电，可以把这些程序选择性关闭。

* 可以离线听写了，不过需要下载语音包，大概700多M。

* 勿扰模式增强，可以在系统选项设置—通知里设置勿扰的时间段、使用模式等，类似iOS7里的勿扰模式。

* 平滑滚动继续增强，无论是在Safari、Finder、iTunes里上下滚动都丝般柔滑，毫无停顿的感觉，明天我测试一下FPS（每秒显示帧数），看看效果如何。



大概就这么多吧，最后发现iTunes居然是英文的，我决定暂时不用它来同步手机，避免误伤友军。

下一篇讲讲应用软件的兼容性和新功能，iBook，Finder，Map，双屏等。最后是性能部分（高科技总是放在最后讲）。

品评**OS X Mavericks——**唯快不破（中）

【发布日期　2013年8月13日】

发了上篇之后，很多人发来贺电说，Mavericks不就是达拉斯小牛吗？怎么成冲浪胜地啦。我建议大家去苹果官网看看OS X 10.9的logo，那一片大浪啊……实在不行去维基百科查查，总之，这里提到的Mavericks和达拉斯小牛的一根牛毛关系都没有，请大家暂时忘掉牛，想着浪。



小伙伴们看了昨天的文章之后说了句天津话，这有嘛？我说别急啊，时光都穿梭到2013年了，如果你还认为升级就是改改UI/UE那就太Naive了，事实上这次Mavericks真正动大手术的地方在底层的技术框架上。我们会在性能一章着重描述。今天讲讲应用软件的兼容性和新功能。

应用软件

1、兼容性

一看到这个标题估计就有童靴扔砖头了，不是说所有软件都不需要重新安装么？请注意，Mac君在“所有”之前加了“几乎”二字，行走江湖……你们懂的。

所以，升级完成之后有几个地方还是需要修缮一下的。

首先，JDK没了，这在前几个操作系统升级时是从未发生过的，最多默默的帮你把JDK5升级为6，而这次是默默的把Java干掉了，难道库

克大爷和Oracle的埃里森结下了梁子？总之，Java在OS X上一直充当一等公民的日子结束了，但也没那么惨，当你需要运行Java相关的应用时，系统会自动提醒你安装Java框架，比如你想打开IntelliJ或Eclipse，Mavericks就会问你要装Java么，你说是的，人家也不会为难你，分分钟就帮你装好了。装完之后，所有相关的程序就都可以正常运行了。

Python比Java麻烦一些。Python依然是OS X的一等公民，并且版本升级到了2.7.5，但这个升级导致的直接后果是以前安装的那些库统统都不见了。比如Django、ElementTree、Markdown、html5lib、MySQL-python等，甚至bpython和ipython两个增强型shell也需要重新安装，让人愤愤不平。这还不算，用easy_install安装时出现的编译错误是你躲不开的宿命，这时候是不就想骂娘了？别急这才刚刚开始……

发生编译错误是因为Xcode的command line tools没装，在OS X里玩命令行没有这货是万万不行的。这时你微笑着打开Xcode，找到Preferences－Downloads－Components，发现，command line tools的安装包不见臭！再一次，Mavericks把XCode 4.6.3的command line tools全部干掉了，而且你还找不到单独的10.9的command line tools安装包，当你想装10.8的包时，系统会明确的告诉你版本不合适，想都别想。

在诅咒了这个万恶的强删制度以后，我决定使出杀手锏和Plan B，在命令行输入“xcode-select——install”，这时会弹出一个选项框，问你是去下载Xcode还是直接安装，选择安装，你会发现，总会有解决方案的。



我担心的Parallels Desktop 8和Papers 2都可以正常使用，用来干活的JetBrains系列和Eclipse、Xcode 4.6.3都没什么问题。

Ruby升到了2.0，在经历了这些之后，我决定让别人去填坑，自己则无耻的跳了过去。

其他我用到的几十种软件大部分都能正常使用，兼容性测试到此结束，简单来说，这个预览版是可以投入日常工作使用的！

2、新成员和新改进

iBooks和地图是Mavericks新增的原生软件，尤其是iBooks，在第五个预览版首次发布。

（一）iBooks的功能类似Amazon的Kindle for Mac，支持云同步，支持打开本地文件，可以做笔记和书签，iBooks对应的文件格式是epub，Kindle是mobi。除了普通电子书，iBooks还能够支持多媒体交互式电子书，比如那本著名的《Life on Earth》，在Mac版的iBooks上显示效果相当惊艳。

终于能够在Mac拥有epub书籍的完美体验，我感觉心满意足。当然，这个新版本问题也不少，崩溃、渲染问题、性能等都像最近的地震一样说来就来，期待最终版。

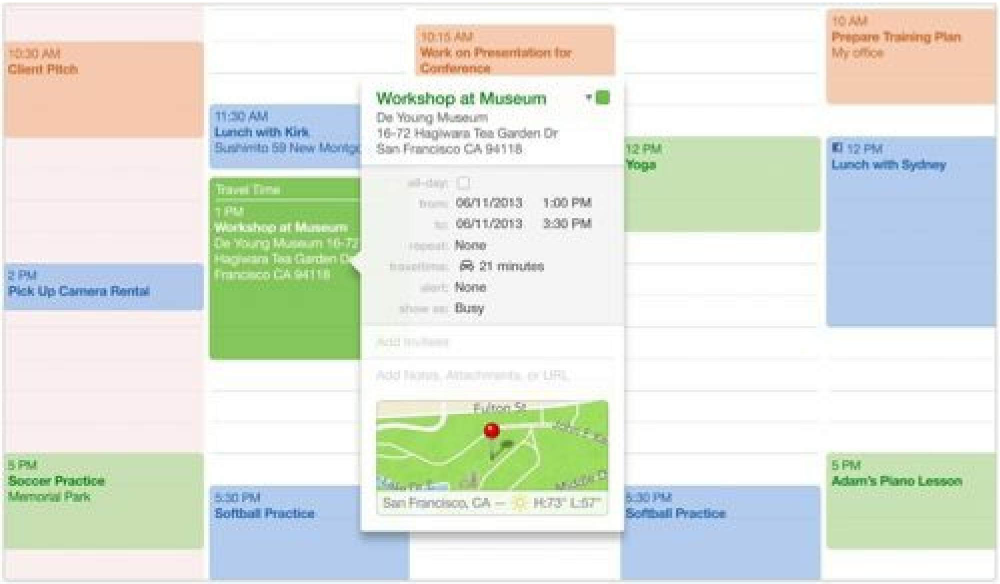




（二）地图就没什么可说的了，就是一个原生的地图软件，大家去看官网介绍即可。

（三）Finder新增了Tab（标签）和Tag（标记）功能，标签这个功能已经被用户吐槽很多年了，在看到苹果迟迟没有动静的情况下，程序员们愤怒的开发出了TotalFinder和XtraFinder这样的软件来告诉苹果，没有你妹一样可以用Tab！Tag的功能可能创意来自互联网的标签云，现在我们除了可以为文件和文件夹标记颜色，还可以打标签，并把这些标记过文件一次显示出来，Mac的文件系统也向着扁平化多维度管理浩浩荡荡的进军了。目前Tag自身的管理比较弱，就一个列表，这要是有50个Tag基本会疯掉，期待改进。

（四）日历和备忘录的UI都重新设计了，尤其是日历的改进，除了设计简约清新之外，周视图可以左右滚动，月视图可以上下滚动，阅读和记录更加方便，事件还能直接与地图位置绑定。备忘录去除了拟物化设计，界面采用了白色和淡黄，字体用了手札体。



（五）Safari的性能、阅读器、边栏、Top Sites都进行了重新设计。阅读器有了更好的阅读体验，基本不需要其他插件了，但取消了底部的设置和导出打印的功能，不知何解。边栏提供了书签、阅读列表和共享链接三个功能面板，和iOS7的Safari一模一样。共享链接会列出你在微博、Twitter上关注的那些小伙伴分享的链接，让你一个链接都不放过。

Safari这次发布的版本是7.0，在高速缓存、页面解析和平滑滚动上做了很大的优化，如果网站和带宽足够，网页几乎没有刷新和加载的感觉，瞬间开启，让你产生一种MD出事了的错觉！长页面上下滚动几乎没有停顿感，我用Quartz Debug测试了一下，FPS能达到60帧，相当强悍。



（六）监视器的UI重新设计、功能增强，为CPU、内存、网络、磁

盘提供了更多图形化的系统信息，界面设计也隐约透出扁平化的味道。

（七）双屏或多屏支持放到最后说，是因为这个功能太赞了。这次多屏的改进并不是扩展或辅助屏幕，而是为扩展显示器增加了一个桌面空间，比如你的Mac原来有三个桌面空间，外接显示器时，系统会默认为你创建第四个桌面空间，并在扩展屏里显示，这个空间是完全独立的，除了没有Dock，其他和主屏完全一样。这样做的好处就是两个屏幕是完全独立的，互不干扰，如果你的机器足够强悍，都可以当两台电脑使用，爱全屏全屏，爱切换切换，互不侵犯，各自为政。对于多桌面空间爱好者来说，吸引力这绝对是致命的。

通知和iCloud Keychain就不细说了，总之让通知和密码保存更加方便，大家届时自己体会吧。

本来想一把写完，结果发现自己太幼稚了，只好把“下”改成“中”，下一篇再谈谈性能和目前存在的一些问题。

品评OS X Mavericks——唯快不破（下）

【发布日期　2013年8月14日】

本来触动我写这个系列的原因就是Mavericks的快，所以起名唯快不破。结果拉开架势写了两篇都还没有写到这个快字，真够失败的。今天的终结篇将主要介绍Mavericks是如何做到“多快好省”。



更正1：Java从Lion开始就不是OS X的一等公民了，而是按需安装，但是之前Lion和ML升级时不会干掉原有的JRE，所以Mac君没有觉察，这次他们干得更彻底一些。

更正2：双屏显示的时候，在扩展的桌面空间里也可以出现Dock，具体操作方式是鼠标在屏幕下方滑动两次，就会呼出该屏幕的Dock。但是这个功能还不稳定，经常发生呼不出来的现象，等正式版吧。

更正3：Ant被干掉了，Maven保留着。看来自古都是“但见新人笑，那闻旧人哭”。依然想使用Ant的童靴可以执行以下脚本安装：

brew install https://raw.github.com/Homebrew/homebrew-dupes/master/ant.rb

性能：唯快不破

一个操作系统除了功能性和安全性之外，考虑最多的就是迅疾如风的响应速度和持久的续航能力，这次Mavericks的升级使用了多种技术保证了OS X的快和节能。

（一）**Kernel**

OS X的内核基于FreeBSD和Mach3.0构建，可以通过32位或64位内核启动系统，并充分发挥32位或64位应用程序的运行效能。在64位内核的情况下，Mavericks经测试可以最大支持128G的物理内存，这特么就是个服务器版本啊。

（二）**Compressed Memory**（内存压缩）

内存压缩技术是Mavericks新增加的底层技术，这货彻底提升了整个系统的使用容量和响应速度，也就是说你可以开启几十个应用程序，在联动内存超过80％，物理内存使用几乎达到100％的时候，依然可以像整个系统只打开了一个程序那样，毫无顿挫感的使用每一个打开的程序，这种美妙的感觉让人想想都要流口水的。

具体原则就是永远去满足最需要内存的程序，当内存告急的时候，系统就会根据时间顺序去寻找那些不活跃程序的内存，将其压缩到一半大小，然后把内存倒出来给最需要的程序。当这些被压缩了内存的程序活跃起来的时候，系统会以最快的速度释放内存，让你切换程序毫无顿挫的感觉。

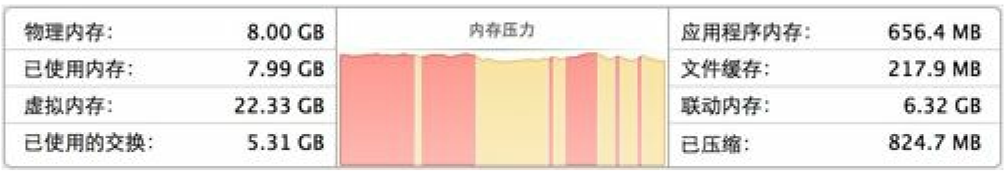
这就像一个家族企业，哥哥姐姐弟弟妹妹各有各的生意，有一天大哥说要干一票大的，这时候其他人就会把手里的闲散资金贡献出来给大哥用，当有其他人也想拓展生意时，也是一样的路数。资源虽然有限，但总能让最需要最活跃的人获得足够的资金。当然前提是活跃的人数是有限制的，都活跃了这个游戏就没法玩了。

Mavericks的内存压缩采用了WKdm算法，这个算法巨牛无比，压缩和解压缩可以说唯快不破。我查了上面提到的论文，这算法也有年头了，不知道为毛现在才翻出来使用。这种压缩技术比硬盘Swap交换技术要快，即使是SSD硬盘也无法比拟。内存压缩的使用，在增加内存使用效率的同时，还减少了硬盘的读写次数和CPU的占用率，另外，与传统的虚拟内存技术不同，这玩意还能充分利用多核CPU的特性，实在是牛到逆天。

所以，大家如果总是把创新定在UI/UE的变化上是图样图森破的，没有技术底蕴都是昙花一现。

为了测试内存压缩的效果，我开启了三十个左右的GUI程序，包括

各种浏览器、阅读器、办公软件、开发工具（IntelliJ、Eclipse、Xcode和模拟器），其中Safari和Chrome分别打开了几十个标签页，整个系统什么感觉都没有，运转如飞。于是开始出大招，打开Parallels Desktop，开启了三个Linux和一个Win7，整个系统的内存变化如下图所见（见网站），红色部分表示内存压力高的时候，也就是我逐一打开四个虚拟机的阶段。这时系统终于有反应了，程序切换略有卡顿，但是依然不影响独立程序的使用。关掉一个虚拟机之后，这个系统又恢复如初了，性能远超上一代的10.8ML。



（三）**Power Efficiency**（电源效率）

Mavericks中的电源技术是根据现代处理器的功能和程序需求构建的，可以有效的延长电池的续航时间，简单来说就是当你的电脑在使用电池供电时，系统会把后台程序的功耗降到最低，尽可能只帮你处理激活窗口的程序，并根据CPU的具体使用状况尽可能让各处理器处于idle状态，一旦有需要再满负荷运行。比如我现在正在打字写文章，CPU和硬盘都会处于低功耗的状态。系统对大功耗的后台程序做了严格的控制，并且会通过电池状态告诉你哪些程序是不合格的，以方便你随时干掉这些程序。

很牛的技术，但苹果没说怎么实现的，杯具！



（四）**App Nap**（程序挂起）

这个技术叫程序挂起也好，程序休眠也好，事实上是和Power Efficiency结合使用的。系统会把你暂时不用的程序，或者不可见的程序迅速而强硬的置于休眠状态，比如你开了三个Safari窗口，每个窗口开启了30个标签页，一共访问90个网址，但是你只能同看一个或几个页面，这时候其他标签也都会被置为挂起状态，当你进行标签也切换时再迅速唤醒。并且在你访问Safari的时候，其他不可见的程序都会被置为挂起状态，限制程序向CPU发出中断请求，降低磁盘和网络I/O。同时，App Nap技术还降低了UNIX基础进程的优先级，以保证系统的低功耗。

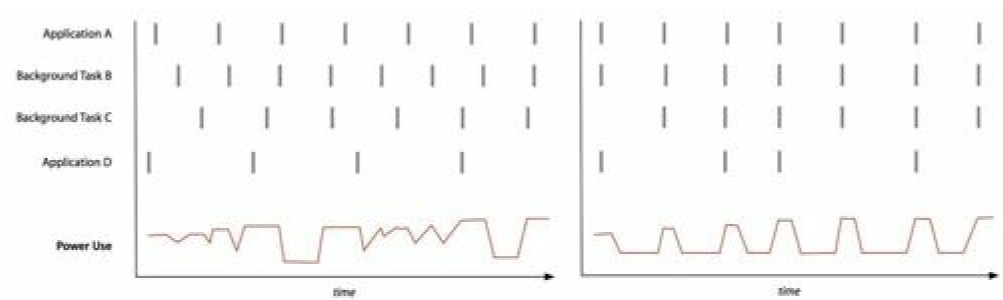
写过程序的都知道，这基本上就是一种懒加载技术，使用的时候才加载，但最神奇的是，系统在做了这么多限制之后，一旦某个程序需要处于激活状态，系统就会开足马力将其唤醒并提供足够的资源，让用户完全感觉不到有休眠和挂起这回事，这让我个人感觉非常悲伤，因为完全不知道人家是怎么实现的。

人比人死，货比货扔，没啥说的。

（五）**Timer Coalescing**（计时器聚合）

最后说说计时器聚合的技术。传统的应用程序在运行的时候向CPU发请求是杂乱无章的，这就搞的CPU很烦，因为永远不知道什么时候能休息，刚眯一下会就可能被叫醒的程序员你们感受下。

比如一位忙碌的程序员，几十位上级轮番上阵的向程序员要资源要成果，而且不分场合不分时间不分类型，长此以往，代码必将写无可写，因为程序员已经疯掉了。这时候就需要一个秘书出现来梳理这些请求，定时定点分门别类的处理，这样程序员的工作就轻松多了，有了计划性，还能得到休息。计时器聚合就是CPU的秘书，它会把请求进行梳理，通过合理的时间管理保证请求的有序性和间歇性，这样CPU就会有更多的低功耗休息时间，具体的处理示意图如下，表示处理之前和之后的CPU时钟对比。同样，用户完全感受不到这位秘书的存在。



好，以上内容根据苹果提供的OS X Mavericks Core Technologies Overview演绎而得，如有雷同，必是他人抄袭。

遗留问题

目前这个DP5的版本已经达到工作机可用的程度，但问题也不少，如下：

1. 外接显示器启动Mac的时候你会看到左右两个灰灰的界面，如果你一直等下去的话，夏去秋来也不会有结果的。拔掉外接插头，你就能看到那张熟悉的脸（登录界面）。

2. 多屏的情况下鼠标经常会失去焦点，如果你发现某个页面怎么也无法滚动时，不要怪罪你的食指，一定是有另一个程序在后面骂娘，“老子已经在后面归隐多时，为毛还要被反反复复的拖来拖去？”

3. 总体来说，Mavericks的UI并没有太大的变化，但是时而扁平时而立体的效果，还是容易让人产生困惑，我们期盼两岸尽快统一起来。

4. 程序的稳定性仍需加强，比如Safari、iBooks等，每天还能崩溃一两回，iBooks的性能和渲染也是问题，丫经常把自己搞得很慢。日历也经常出问题，你会发现周历和月历突然不能滚动了，或者用着用着事件的颜色被偷偷的修改了，总之，凡做过大手术的程序，都不稳定，Xcode5.0没敢升级，不作死就不会死。

希望下一个预览版能够解决这些问题。

另外，应部分童靴的要求，我把Mavericks 10.9DP4完整镜像包和DP5的升级包的下载链接都放到了这篇文章的后面，感兴趣的可以去试试，尝试新鲜事物永远是保持内心鲜活的动力之一，但这毕竟是给开发者的预览版，记住两点：

1. 先备份

2. 杯具了就别来找我

苹果的语言

【发布日期　2013年4月10日】

最近连续被问到学习Objective-C的问题，比如我原来学的是C或C++，学Objective-C行吗？这门语言语法怎么那么古怪呢？现在学Objective-C有前途吗？

好吧，今天简单说说苹果的当家语言Objective-C。



首先说明一下，技术这种东西很难说学哪个更有前途，而且以前有一期也说过，如果你准备以技术手段谋生或实现你的理想，基本不可能只懂一门语言。就目前来看，Objective-C似乎是薪酬较高的语言，但是这个阶段迟早会过去的，就像当年的Java一样。一门语言只有掌握的足

够深入，才有机会探讨它的前途和钱途。

想要看得远，必须站得足够高。吐槽完毕！

Objective-C是C语言的扩展，设计思路上借鉴了Smalltalk的面向对象和消息机制的思想。从我个人使用过的面向对象语言来看Obj-C是对消息传递支持的最彻底也最显式的。Objective-C的类中定义的方法都是消息传递，而且类和消息之间是运行时绑定的，运行时编译器把消息发送转换成对objc_msgSend方法的调用。其它的C++、Java、Python、Ruby都体现得不明显，更倾向于对象的封装和抽象。

Objective-C和C++基本上是两门语言，没有太大的关系。Objective-C本身是静态语言，编译后就是机器码，执行效率很高，但引入了很多类似Python、Ruby的动态特性，像动态类型推断，id，selector，block等特性，所以又非常灵活。用惯了Java或C++会觉得Objective-C的语法很怪，但如果你用学习新事物抛弃旧思路的方式去学习这门语言的话，你会很快爱上它的消息式的编程风格，加上XCode，无论是写Mac应用还是iOS应用，都会非常得心应手。

关于苹果为什么采用Objective-C的问题，说明一下，其实不是苹果采用了Objective-C，而是乔布斯创建的Next公司的操作系统NextStep采用了Objective-C作为原生语言。

Objective-C在计算机编程语言中有着不短的历史，80年代初Brad Cox和Tom Love发明了Objective-C，1988年乔布斯的Next公司获得了这门编程语言的授权，并开发出了Objective-C的语言库和NextStep的开发环境。NextStep是以Mach和BSD为基础，Objective-C是其语言和运行库。后来的事大家都比较清楚了，苹果买了NextStep，乔布斯回归苹果，NextStep也成了Mac OS X的基础。以后发展越来越好，Objective-C反而成了苹果的当家语言，现在基本上是苹果在维护这门语言的发展。

随着苹果的APP帝国不断壮大，这门语言也得到了长足的发展，从1.0到2.0，从面向对象的C语言扩展，到内存引用计数管理，属性管理，引入块的概念，实现自动引用计数，优化编译器，简化语法等等。Objective-C在2011年和2012年分别获得了TIOBE评选的年度语言，目前排名第四。

苹果CEO库克在2012年的WWDC大会上宣布，苹果已经为全球开发者支付了超过50亿美金的分成收入，时至今日，估计已经远远超过50亿了，对于开发者来说，这是一门能够独立创富的编程语言。

还有一点不能不提，第一台万维网的Server就是一个叫蒂姆·伯纳斯-李的大牛在NextStep上写的，包括浏览器。所以，咱们得感谢Objective-C，要不然还不知道互联网会发展成啥样呢……

说说我和Mac（一）

【发布日期　2013年4月19日】

30岁之后，时间仿佛开闸的河水一样滚滚而去，感觉自己浪费的时间太多。我们不得不承认，先知先觉的人会比我们领先10年甚至更多的身位。进入二十一世纪，世界迎来了信息时代，人才竞争尤其激烈，与其他人感受不同的是，我觉得中国涌现出了很多非常厉害的年轻人，很多人年纪轻轻就成为某个领域的行家里手，端的是少年英雄。不过放眼未来，无论现在的你是年少成名还是大器晚成还是默默无闻，都需要不停的奔跑和追赶。

感慨完了，说说今天的内容，很多读者让我讲讲自己的经历，非常惭愧的是，工作十余年并无可圈可点之事，实属籍籍无名之辈，谈之无物。倒是可以讲讲我对Mac的些许认知，中间也可以穿插讲点有意思的事情。今天是第一篇。

说起Mac，还得从2001年说起，那是我第一次接触Mac电脑，当时我在洪恩软件开发了一套叫做数字校园的软件系统，由于与一家厂商合作，需要把我们的软件移植到Mac Server上。


Mac OS X
Version 10.2
Jaguar

软件是BS架构的，基于JDK1.3构建，由于一直在linux上编程（当时Java几乎没有像样的IDE，Eclipse、NetBeans、IDEA等后来如日中天的工具，有的刚刚起步，有的还在孵化），所以Mac基本上是被我们当做Unix用的，印象中移植并没有太大的工作量，细节也不记得了，反正当时Mac对我来说就是一个Unix Server，以至于我现在完全不记得当年那台Mac服务器是什么样子了。

后来有另外一个组的兄弟要做音乐，公司专门给他配了一台Mac Pro，价格相当昂贵，那个兄弟估计也是没用过好东西，护的紧，基本不让我们这些土鳖程序员靠近，那时候Mac OS X已经告别了9，进入了10。如果记忆没有失误的话，那个系统用的是Mac OS X 10.2Jaguar，其华丽的界面让一直用土土的windows的程序员口水留了一堆，但当时我们已经被Liunx下各种华丽但不实用的GUI伤透了心，像GNome，KDE基本都是浮云。于是苹果在我眼中就是个酷酷的操作系统，界面优美，适用于图形图像视频制作，价格贵的离谱，用来工作娱乐神马的，基本上是天方夜谭了。

现在留下的印象就是头发杂乱的流浪歌手，谈着吉他，安详的坐在Mac Pro前调音和谱曲的画面。那个兄弟叫老郭，专门为软件做音乐，经常对我们这些不懂艺术的程序猿说，嗯，你们都是土鳖，就知道编程，多无聊。当时我想，这话反过来说，也成！

我在洪恩软件一共工作了3年半的时间，这段经历让我的能力和见识得到了长足的成长。期间做过互联网（没错，就是传说中的第一波互联网神奇泡沫，一触即溃），基于Perl构建洪恩在线网站；做过企业级软件数字校园，基于Java和Jsp技术；做过英语培训软件，基于.Net和C；还管理过儿童产品事业部。不过让人伤感的是，我参与的大部分软件项目都以失败告终，洪恩在线遭遇互联网泡沫，数字校园过于超前，销量一般，英语培训碰到2003年非典，完全无人问津（这部分内容在“缅怀那些沉没的项目”一文中有述）。当时洪恩的主要的业务还是电脑教育、高教、英语、儿童软件等，现金流良好。但安则思变，公司开始尝试其他的业务类型，比如互联网、游戏、企业软件、培训等等，这些项目我基本都参与了，但成者寥寥，这对当时年少轻狂的我来说打击非常大，经常参与失败或无疾而终的项目会让人产生对己对人的怀疑和不自信。

要么留下来继续坚持，要么出走寻找新的挑战，这是很多人当时的选择。有些人选择坚持，有些人选择离开，留下的很多技术人员组成了后来的游戏公司完美时空的技术和策划班底，完美时空2007年在纳斯达

克上市，目前发展得非常好。离开的人，则就各自有各自的生活和精彩。

所以说选择这个动作真的很神奇，人生漫漫征途，到处十字路口，每次选择就把你带向一个完全不同的路和沿途的风景，我们只能慨叹，嗟夫，人无常势，水无常形。

当时我选择了离开，那段时间是1999到2003年。在这段时期里，地球的那一边，乔布斯正在重新整肃苹果，在进行了精兵健身并相继推出了iMac和iPod，苹果正在一步步走向巅峰，只不过还没有看看到这家没落帝国的潜力，在国内，Mac仍然是稀有物品，我也仅仅见过两款。

（待续）

选择Mac

【发布日期　2013年7月16日】

今天去苹果官网上扫了一眼，发现Macbook的价格比我之前购买的优惠了不少，加上有不少朋友让我说说如何选择Mac，那就简单说两句。

苹果目前Mac系列的产品有MacBook Air，MacBook Pro，Mac mini，iMac，Mac Pro。

1、首先来说说MBA，如果追求轻薄便携的话，当然首选Air系列，如果选Air系列，首推11英寸系列，因为11英寸的键盘宽度和13英寸差不多，13英寸只是增加了屏幕高度，所以在输入性上二者并无太大区别，但在移动方面11寸就能落13寸的几条街了。Air系列目前是续航能力最强的笔记本，11英寸款续航时间长达9小时。全系列采用了SSD硬盘，性能无需担心，打开程序文档基本秒开，所以不仅是商务人士，程序员一样可以选择Air，你可以随时随地拿出11寸的Air进行编程、写作、浏览，无需担心热量、重量和续航。同时，如果需要使用大屏编程，接上蓝牙键盘、鼠标和外接显示器，即可打造一台强劲的台式电脑。



27 英寸 iMac

目前MBA全系列不支持Retina显示屏，应该是技术上还没有解决如此轻薄的笔记本如何适配视网膜屏。

2、MacBook Pro是Mac系列里的中坚力量，深得广大人民群众的喜爱。如果你想追求更强的性能、更好的体验、更清晰的画面，那么我推荐你购买配备了Retina显示屏的MBP15，MBP系列里的顶配，也就是传说中的RMBP15，笔记本中的跑车，跑车里的战斗机！如果你选择了足够大的内存和硬盘，可谓一本在手走遍天下都有，画面之清晰性能之强悍让你恨不得贫贱不能移威武不能屈，你可以编程可以作画可以视频可以阅读，你还可以建三个虚拟机并且同时启动，然后搭一个夹杂着私有云和公有云的混合云，什么LVS、Cloud Foundry、Hadoop、NoSql，能用的都给丫用上，当客户问你们的私有云有demo吗，你沉默着指着自己的包包，都在这里，无比的高端大气！

当然RMBP15的缺点是价位较高，虽然已经变得轻薄了，但依然不适合手提斜跨奔波街头，电池的续航能力也是个问题，只能支撑4-6小

时。退而求其次是RMBP13，屏幕、性能、CPU、重量、价位同步降级，也算个上选。旧版的MBP就不推荐了，基本上都是过渡产品。

3、Mac mini分为普通版和Server版，操作系统也分为OS X Mountain Lion和OS X Server，和普通PC不同，mini就是一个铝合金的漂亮盒子，除了一堆接口啥都没有，但是价位低廉，性能强劲，适合学生或在经济性上考虑较多的用户，配上外置键盘、鼠标和显示器，应该是个很好的Mac解决方案。

4、iMac是Mac的一体机，外观看上去就是一台帅到逆天的显示器（本文的配图），其中低调隐藏了主机和各种外设，屏幕分为21.5英寸和27英寸。购买前者还是后者，我觉得主要看你自己的书房和书桌大小，在苹果店看起来不算什么的iMac27，放到自己书房，就特么一个字，大！有了iMac 27，神马双显示器，什么9个Space都弱爆了，你只要把自己的窗口平铺开放到屏幕上就行了，还是那个字，大！如果是穷人家的孩子，第一次用iMac会眩晕的，你会说妈妈我再也不用窗口最大化了……

iMac主要是在工作室或家用，配合一个11寸MBA似乎是不错的选择。

5、最后介绍一下Mac Pro，这货是Mac系列中动力最为强劲的机型，号称战斗机里的宇宙飞船，而且是桶装的……它那忧郁的光泽，浑圆的腰身，神乎其技的性能和存储，还有那基于Xeon E5芯片组的12核CPU，都掩饰不住Mac Pro的出众，它代表了专业台式电脑的未来……

如果你们不信，看看这个：http://www.apple.com.cn/mac-pro/，如果不是专业人士，我不告诉他！

Mac Tips

1．终端输入说英语

说英语时我们当然希望有标准发音。在Mac中不需要字典，直接在终端里输入say yes，Mac就会说英语了

2．Spotlight快速打开程序

很多刚开始使用Mac的用户，一般都知道Spotlight检索功能。事实上用这个功能还可以快速打开程序。通过ctrl+space呼出，输入通讯或cont，都可以找到通讯录这个程序，回车即可打开。

3．Spotlight注释功能定位文件

OS X的文件系统提供了Spotlight注释功能，可以帮助用户更有针对性的定位文件。选中一个文件或文件夹，command+I打开简介，在Spotlight注释功能中加入自己特定的关键词。关掉简介窗口，呼出Spotlight并输入刚才的关键词，可以准确定位到相关的文件或文件夹。

4．使用sips命令批量处理图片

如果你想批量修改一批图片（尺寸、旋转、反转等），但你不会或没有PS，可以使用sips命令高效完成这些功能，例如：

```
#把当前用户图片文件夹下的所有JPG图片宽度缩小为800px，高度按比例缩放
sips -Z 800~/Pictures/*.JPG
#顺时针旋转90°
sips -r 90~/Pictures/*.JPG
#垂直反转
sips -f vertical ~/Pictures/*.JPG
```

更多命令可以用sips -h查看。

5．command+I直接打开邮件

使用Safari浏览网页的时候，如果你想把当前页面通过邮件发送给自己或别人，使用command+I，可以直接打开邮件并把当前网页附加到待发送的邮件中。

6．shift+command+delete自动清空废纸篓

如何快速删除文件和清空废纸篓呢？在Finder中选中文件，使用command+delete删除文件，如果想彻底清除，使用shift+command+delete就会自动清空废纸篓。

7．输入du -sh *获悉目录空间

在Mac下想知道某个目录下各个文件和子目录各占多少空间，不需要一个一个去查看。打开终端，在该目录下输入：du -sh *，结果一目了然。

8．英文自动完成

当使用系统软件文本编辑、Pages、Keynote时，输入英文按esc键，系统会帮助你自动完成单词，比如你想输入brilliance，只需输入brill，按esc键，系统就会出现自动提示。如果某个应用，比如Safari的搜索框里esc是取消输入，那么使用fn+f5也可以达到这个效果。对于常写英文文档的人比较有帮助。

9．文件操作

在Finder中打开文件使用鼠标双击或command+O，和Windows不一样的是，选中文件回车是对文件重命名，而不是打开文件。

10．显示隐藏文件

在终端里输入ls -a，可以显示该目录下的隐藏文件。在Finder中输入shift+command+.可以显示隐藏文件，想恢复原来的设置，再输入一遍shift+command+.即可。

11．利用你的触发角

OS X系统为用户提供了强大的Mission Control功能，今天为大家介绍其中的触发角。打开系统偏好设置-Mission Control-触发角，就可以对屏幕的四个角进行设置了。比如把左上角设置为将显示器置为睡眠状态，当我们暂时离开电脑时，顺手把鼠标移到左上角，屏幕就变黑了，非常方便。

12．维护你的Mac

Mac的OS X是一个使用起来非常简单的操作系统，一般情况下不需要装杀毒工具，大部分程序安装都非常简单，直接把后缀为App的程序拖进应用程序文件夹就可以了。但是，当你在使用系统时如果发现出现异常，那么就该进行日常维护了。

打开磁盘管理，选中你的系统盘，点击“修复磁盘权限”，对磁盘权限进行检查和修复。完成之后还可以手动执行维护脚本：

```
sudo periodic daily
sudo periodic weekly
sudo periodic monthly
```

也可以一次全部执行：

```
sudo periodic daily weekly monthly
```

一般执行完这些操作后，你的Mac就会充满活力，继续上路。这些操作可以定期执行。

13．窗口按应用程序成组关掉取消同一个程序窗口重叠

在Mission Control设置中把“使窗口按应用程序成组”关掉，Mission Control的行为就会跟10.7以前的expose一样，不会把同一个程序的多个窗口叠在一起。对经常一个程序开很多窗口的程序员来说很有用。

14．截图

OS X提供了非常方便的截图工具，你可以随时随地截取屏幕画面。

shift+command+3：全屏幕截图；shift+command+4：通过鼠标选取截图。

截取的图片默认存放在桌面上，以时间命名。

系统默认截图格式是png，你可以通过如下命令修改截图文件类型，例如：

```
defaults write com.apple.screencapture type -string JPEG
```

15．推荐几个有用的小工具

TotalFinder：Finder的增强插件，Finder的插件，为Finder增加多标签（类似Chrome的多页签）、双面板、UI设置等功能。收费软件。

Breeze：窗口管理软件，Optiong+1/2/3分别对应最大化窗口/左半屏幕窗口/右半屏幕窗口。收费软件。

Trillian：整合了MSN，GTalk，Twitter等，表现稳定，用户体验也不错。免费软件，可以从App Store直接下载。

smcFanControl：风扇控制软件，免费。OS X对风扇控制不敏感，CPU温度很高时才会增加风扇转速，那时机器表面已经比较热了。用这个软件可以自由控制风扇转速。夏天空调屋里一般3000-4000转就够了，冬天一般不需要开启。

16．Mac的原生输入法

我在Mac下曾经使用过很多输入法，包括百度、FIT、搜狗等，这是因为之前Mac的原生输入法太不给力了。不过现在的版本已经有了很大的改进，慢慢的也变成了常用输入法之一，今天就为大家介绍一些Mac输入法的操作技巧：

·中英文混合输入，输入中文的时候，打开caps lock键，可以直接输入英文，关掉又切换回中文
·选词，通过－＋号可以切换字或词，通过［］可以展开候选词列表并进行切换
·打开输入法偏好设置，可以设置自动校正模糊音
·用'可以进行手动分词，比如fang'an（方案）
·使用shift +6可以输入表情符号，比如（☆_☆）凸^-^凸

用习惯了，你会离不开这个输入法的……

17．Safari的标签

Safari是我在Mac上最常用的浏览器，Chrome也不错，但我更偏爱Safari。今天为大家介绍一下这个浏览器的标签使用。当你想在新的标签页打开网页时，只需要按住command键，点击链接即可。使用Multi-Touch手势在标签页中切换。在触控板上，双指开合即可显示你打开的标签页。在标签视图中，双指轻扫可浏览不同标签页。通过shift+command+左右方向键，可以快速在Safari中打开的标签中进行切换。

18．监控Mac的运行状况

·top

打开终端输入top，可以显示目前系统的进程情况、CPU使用情况、内存使用情况、磁盘使用情况和进程的详细列表等信息，输入？会显示帮助信息，参考帮助你还可以自定义top显示的信息，输入q退出监控界面

·htop

htop是更聪明更高级的top，虽然不是Mac原生的，但安装非常方便。打开终端输入：sudo port install htop，命令结束就安装完成了。然后键入htop，你会看到一个更丰富的彩色的top，多个CPU、内存统计、uptime，更详细的进程信息。参考界面最底部的帮助信息还可以对进行排序、展开、Kill。输入q退出监控界面

·系统的活动监视器

这个非常适合不喜欢终端的用户。从应用程序-实用工具可以找到活动监视器，打开后你会发现很类似windows下的任务管理器，相信这个不需要给大家介绍了

19. 批量复制文件

例如你在一个目录下林林总总放了几百个文件，有图片有pdf有zip有doc等等，你想把后缀为png、jpeg、gif的图片复制到另一个文件夹去，最简单的方式是什么？

不是通过搜索把这些文件找出来，再全选复制到另一个文件夹下。而是进入该目录，执行这样一个命令：

```
cp *.png *.jpeg *.gif /destpath
```

如果想剪切，就把cp改为mv

20. 程序切换

在OS X中程序切换可以通过command+tab进行，command+tab进行顺序切换，command+shift+tab进行逆序切换，功能类似Win7的alt+tab。

OS X还提供了同组程序的切换，比如你打开了多个预览程序阅读pdf，你想在这些pdf之间切换阅读，这时候就可以使用command+`（esc下面的键）进行同组程序切换。

21. 远程拷贝

OS X提供基于ssh的远程拷贝命令scp，这个命令大部分linux和unix系统都会提供，使用该命令可以非常方便的在两台机器之间安全的复制文件，具体命令：

```
scp ./testfile.txt username@10.10.10.22:/tmp
```

回车后会要求你输入username的密码，只会将当前目录下的

testfile.txt复制到另一台机器的tmp目录下。

scp username@10.10.10.22:/tmp/testfile.txt./

从远端复制到本地。

22．OS X中的ftp

这个问题有订阅者问过，总结一下，以下三种方式就够用了：

·直接在命令行使用，打开终端输入ftp anonymous@ftp.mozilla.org，或者使用sftp通过ssh完成ftp的功能，例如sftp user@10.10.10.11。
·使用第三方工具，比如FileZilla，用法和windows类似。
·利用OS X原生ftp工具，从Finder菜单栏中进入“前往-连接服务器……”，输入FTP服务器地址（如：ftp://ftp.mozilla.org）点击地址栏右侧的+号按钮可以将当前地址加入“个人收藏服务器”点击“连接”按钮，按照提示进行身份验证成功后即可连接到FTP服务器。

23．备份

OS X提供了非常方便的备份工具TimeMachine（时间机器），我第一台Mac用的操作系统是Leopard，后来升级到Snow Leopard—Lion—MountainLion，换新机器，但从未重装过系统，这对于Windows系统来说是不可想象的，这都得益于时间机器。我个人每周会备份一次，如果你觉得自己资料非常重要，可以每隔几小时备份一次。具体的用法我就不介绍了，可以参考官方介绍。

24．inode和history

·inode：Mac的文件系统和windows完全不同，文件所需信息都包含在这个inode（索引节点）里。每个文件都有inode，文件系统用inode来标识文件。简单来说就是inode包含了文件的元数据信息，文件名、文件内容并不包含任何控制信息。inode是unix/linux系列文件系统设计的核心，有兴趣的童靴可以上网查阅相关资料。对于普通用户用来，最直观的表现是，在Mac里，你可以对正在使用的文件改名，换目录，甚至放到废纸篓，都不会影响当前文件的使用。
·history：打开终端输入history，所有的历史命令都会显示出来，想找某一条执行过的命令，还可以这样：

history |grep apache
找到左边的命令编号（例如1001），在终端输入
!1001
就可以执行原来那条命令了。

25．Go2Shell

通过Finder浏览文件的时候，常常需要在浏览的文件目录中打开终端进行操作，Go2Shell能够自动做到这一点。

从App Store下载这个免费软件。下载完成后从应用程序文件夹把Go2Shell拖到Finder工具栏上，然后随便进入一个目录，点击Go2Shell图标，即可打开终端进入该目录。

Go2Shell支持原生终端、iTerm2和xterm，在终端输入

```
open -a Go2Shell——args config
```

即可进入配置界面，选择你喜欢的终端。

26．Safari的阅读器

Safari的阅读器是浏览器创新之一，在Safari之前，没有其他浏览器提供过这样的功能。当Safari发现结构优良的网页文档时，就会在地址栏右侧显示“阅读器”，点击就可以进入简洁的阅读模式，通过shift+command+r也可以进入。

阅读器已经提供了良好的网页阅读体验，对于分页文档甚至能够自动翻页阅读，但是我们还可以更进一步。比如我就觉得阅读器太窄了，视野不够宽阔。有类似需求的童靴就可以通过safari的扩展插件CustomReader进行个性化定义。

从这里下载CustomReader，双击可安装。安装之后到任何一个支持阅读器的网页，按下shift+command+r激活阅读器，再用ctrl+r调出配置页面，就可以配置你自己独享的个性化阅读器了。

27．Remote Desktop Connection for Mac

很多读者询问如何在Mac中通过远程桌面连接到Windows，这次统一答复一下，微软提供了专门的Remote Desktop Connection for Mac，免费，下载链接：http://www.microsoft.com/mac/remote-desktop-client

28．文档的版本控制

经常使用Keynote、Pages、Numbers和原生文本编辑器的用户，可以尝试使用文档的版本控制功能。对于经常编写文档的人来说，这个功能非常有用，大家可能没有注意到，当你把鼠标移至文档标题的时候，会出现一个小箭头，下拉可以看到浏览所有版本的选项，点击进入该文档的时间线，界面与Time Machine一模一样，你可以非常方便的找到任

何一时间点你编辑过内容，也可以随意恢复到任何一个版本而不会影响其他版本。非常酷的功能，并且好用。

29．如何快速发送带附件的邮件

在Windows我们可以右键点击文件发送到邮箱即可发送带附件的邮件，OS X也有类似功能，只不过叫共享。右键点击要发送的文件—共享—电子邮件即可。

30．如何快速创建便笺

便笺是我们很常用的功能，可以把一些临时性的文字内容贴到桌面上，大家是如何做的呢？复制文字，打开便笺程序，新建便笺，粘贴文字！Too young too complicated，我们只需要选中文字，然后shift+command+y，就行了。

31．Mac的通用快捷键

这部分内容之前陆续介绍过，但还是有童靴希望有个汇总，基于二八原则，我把最常用的快捷键罗列一下，对于非开发者，应该够用了

Command+Tab	任意情况下切换应用程序-向前循环
Shift+Command+Tab	切换应用程序-向后循环
Command+Delete	把选中的资源移到废纸篓
Shift+Command+Delete	清倒废纸篓（有确认）
Shift+Option+Command+Delete	直接清倒废纸篓
Command+~	同一应用程序多窗口间切换
Command+F	呼出大部分应用程序的查询功能
Command+C/V/X	复制/粘贴/剪切
Command+N	新建应用程序窗口
Command+Q	彻底退出当前应用程序
Command+L	当前程序是浏览器时，可以直接定位到地址栏
Command+"+/-"	放大或缩小字体
Control+Space	呼出Spotlight
Command+Space	切换输入法

对于最后两个快捷键，我个人比较习惯Control+Space切换输入法，所以做了自定义的配置。

32．终端命令open

我们之前介绍过如何在Finder中浏览文件时进入当前目录的shell界面，那个插件叫做Go2Shell。当然我们也会有在shell下打开当前目录的Finder的需求，运行如下命令即可：

```
open .
#当然open也可以打开其他目录，比如
open /Users

#open还可以直接打开文件，打开程序，指定程序打开文件，打开网址等等，例如
open a.txt
open -a Safari
open -a TextMate a.txt
open http://news.sina.com.cn
```

33．CatchMouse自定义快捷键

介绍小软件——CatchMouse，可以自定义快捷键快速在多个显示器内切换鼠标，非常方便，链接附上：

34．激活窗口

如果你在一个屏幕内打开了多个程序，除了当前激活的软件窗口，你还想看看其他窗口的内容，这时你直接点击其他窗口的话，原来的窗口就可能被遮挡或消失。如何保持原来的窗口一直处于最上层呢？非常简单，拖拽其他窗口的时候按住command键即可，原来的窗口会永远在最上面。

35．文件检查器

在windows中大家经常选中多个文件，右键-属性可以查看这些文件的大小。在Mac里同样的操作（选中多个文件，右键-显示简介），弹出的是各个文件或文件夹的简介，这让很多童靴困惑不解。其实我们只要在点右键的同时按住option键，显示简介就会变成显示检查器，点击显示检查器即可查看和操作批量文件。

另外，我还经常用这种方式浏览图片，比如选中多张图片，按住option键点击鼠标右键，选中“幻灯片显示xx项”，就可以全屏浏览图片了。

36．屏幕放大镜

有时我们需要放大屏幕做一些精细的操作，ctrl+鼠标滚轮可以实现这一效果，如果你是键盘控，用option+command加上加减号也可以实现。

37．语音识别

Mountain Lion增加了语音识别的功能，具体的设置在系统偏好设置-听写与语音，你可以设置听写语言、呼出窗口的快捷键等等。我采用是默认的快捷键，连续按fn键两次即可呼出语音识别窗口，这时候你就可以对Mac说话了。如果你想让Mac把你说的写下来，最好打开一个的文本并让光标处于可编辑状态。注意，该功能需要联网。

38．time命令

如果你想知道在终端执行的某个程序耗时多久，对CPU等的使用情况，可以输入：

```
time python fib.py
```

输出结果：

```
python fib.py 0.02s user 0.02s system 50％cpu 0.094 total
```

39．特殊字符输入

美元，shift+4
美分，option+4
英镑，option+3
人民币，option+y
欧元，shift+option+2
波折号，option+-或shift+option+-
省略号，option+;
约等于，option+x
度，shift+option+8
除号，option+/
无穷大，option+5
小于等于，option+,
大于等于，option+.
不等于，option+=
圆周率Pi，option+p
正负，shift+option+=

平方根，option+v
总和，option+w
商标Trademark，option+2
注册，option+r
版权，option+g

40．OS X三指轻拍查找功能

OS X提供了三指轻拍查找的功能，什么意思呢？把光标移到一个单词上面，无需选中，三指轻拍，系统就会弹出词典显示相关单词的释义，非常方便。该功能可以在系统偏好设置-触控板里进行设置。

41．F.lux调节屏幕色温

推荐一款免费小软件F.lux，这个软件功能类似系统的亮度自动调节，不同的是它调节的是屏幕的色温。该软件能够根据时间来调节屏幕色温以达到保护眼睛的目的。有数据表明，4600k到5000k的暖色有缓解眼部疲劳的作用。下载地址：http://stereopsis.com/flux/

第一次打开应用，需要输入当前城市名称，搜索定位用户的当地时间。F.lux将根据日出及日落的时间来调节色温。在日出日落期间，屏幕色温和平时一样，对于RMBP来时就是6500k；日落之后，F.lux会逐渐地暖化你的屏幕。具体色温可以自定义。

我用了之后感觉还不错，大家可以试用下。

42．输入pmset noidle，Mac不休眠

如果你想离开电脑一段时间，又不想让电脑进入睡眠状态，有个简单的命令可以帮助你做到这一点。在终端中输入：pmset noidle，即可。只要该命令一直运行，Mac就不会进入睡眠状态。关掉终端或ctrl+c可以取消该命令。

pmset是OS X提供的命令行管理电源的工具，其功能远不止于此。

pmset -g，查看当前电源的使用方案
sudo pmset -b displaysleep 5，设置电池供电时，显示器5分钟内进入睡眠
sudo pmset schedule wake "02/01/13 20:00:00"，设置电脑在2013年2月1日晚8点唤醒电脑
……

感兴趣的可以使用man pmset查看详细信息。

43．网络共享

Mac提供了非常简单易用的Internet共享功能，可以作为一个轻量级的家庭无线路由使用。只要你的Mac能够上网，那么phone和pad等设备就都可以通过wifi共享Mac的网络，实现无线上网。具体的设置非常简单，打开系统偏好设置-共享-互联网共享，选择共享源（网卡或AirPort），并设置wifi的名称密码安全级别等属性，最后勾选左侧列表的“互联网共享”，根据提示操作即可。

这是一个我曾经认为大部分用户都知道的功能，后来发现几乎很少人使用或会用。

44．快速查看

OS X提供了非常方便的预览文件内容的功能。在Finder或桌面上，选中一个文件并按空格键，系统就会弹出预览界面。对于很多文件我们仅仅使用快速查看功能就可以浏览文件内容了，比如iWorks的keynote、pages、numbers，微软Office的文档，pdf，图片，视频，各类文本文件等等。

除了在Finder和桌面快速查看文件，我们还可以快速浏览邮件的附件。打开邮件程序，找到一个带有附件的邮件，选中附件并按空格键，就可以快速浏览附件内容。

我们还可以在终端操作的时候使用这个功能，例如qlmanage -p文件名，系统就会弹出快速查看窗口。

45．显示桌面

我们下载文件或临时文件经常会放到桌面上，在Windows里通过alt+d或点按显示桌面的图标即可，在Mac里如何实现呢？

有两种方式，都很方便，第一种是四指划开，该功能可以在触控板里设置。还有一种方式是通过快捷键command+F3，即可实现移开程序显示桌面的功能。

当我们想把桌面的文件放入某个程序（比如当做邮件附件）时，可以配合command+tab实现。用鼠标拖动桌面文件，command+tag切换程序，然后把文件拖入该程序即可。

46．应用程序的安装和卸载

OS X中的应用程序和OSGi中使用的Bundle类似，都是把配置文件和程序封装在一个包里。对于普通用户来说，你在Launchpad中看到的所有程序都像一个图标，但这个图标不是Windows中的快捷方式，而是

封装好的Bundle，从程序角度而言这是一个文件夹，对普通用户来说，知道点这个图标运行程序就行了。这种设计方式使得OS X中95％以上的软件的安装变得十分简单。如果你是从Windows转过来的话，你会认为安装和卸载简单的令人发指。安装程序就是把XXX.app拖进/Applications（应用程序文件夹），卸载就是把程序从该目录删掉。好吧，你可以这么理解，OS X中95％以上的软件都是Windows中的“绿色软件”。

47．磁盘映像

磁盘映像类似Windows中的iso，不过文件后缀为dmg。磁盘映像可以直接挂接到OSX中，其表现形式就像是磁盘分区。双击文件可以直接打开，打开后在Finder左边栏的设备中可以找到挂接好的磁盘映像。dmg是Mac下最常用的文件组织方式，几乎所有的安装程序都是以dmg方式发布的。一般情况下安装程序就是打开相关程序的dmg文件，里面有一个app和应用程序文件夹，把app拖入应用程序即可。另外我们也可以使用磁盘工具把dmg里的文件恢复为真正的硬盘文件，也可以制作dmg文件。

48．复制目录下文件名列表

如何复制某个目录下所有文件的文件名列表呢？非常简单，command+a，command+c。然后打开一个文本编辑器（比如TextMate），command+v即可。

49．多点触控手势

当我们用Safari浏览网页时，经常想回到之前浏览过的历史页面，使用多点触控手势可以非常容易直观的实现该功能。打开Safari浏览多个页面，然后使用双指左右轻扫，可以来回切换浏览页面。

另外，如果你在浏览时不小心关掉了一个标签页，使用command+z可以恢复最后关闭的那个标签页。

50．OS X的预览程序

OS X的预览程序可以打开各类图片和pdf等类型的文件，当你想查看某个图片或pdf的细节时，没必要用command+±来缩放整个文件，使用`键可以呼出放大镜，细节一览无遗。

51．显示隐藏桌面内容快捷键

我们经常会在桌面上堆满文件夹和文件，有时候会很方便，有时候会觉得很乱。其实我们可以通过以下命令来决定什么时候显示，什么时候隐藏。

```
chflags hidden ~/Desktop/*                //隐藏桌面内容
chflags nohidden ~/Desktop/*              //显示桌面内容
```

如果觉得输入麻烦，用TextExpander或Alfred设置成snippet即可。

52．按住option的快捷键

OS X设置了一些快捷键用来快速打开显示器、MissionControl、键盘、声音等系统设置，具体是什么呢？你只要按住option，轮番把键盘最上方的那排键试一下就知道了，一般人我不告诉他。

53．Space（空间）

使用OS X，我们可以充分利用系统提供的多个Space，把不同的程序放到不同的Space，让我们的系统更有扩展性。如何增加Space呢？四指上推，在桌面的最上方会出现当前的Space，把鼠标移到Space列表的右侧，会出现一个带＋号的空间，点击加号，即可增加一个Space。

那么如何把某个程序固定在某个Space打开呢？在某个Space打开程序，在Dock中找到这个程序图标，鼠标长按会出现一个菜单，选项-分配给，选“这个桌面”，下次再打开这个程序，就会自动进入设定的Space。

Space的排列方式可以在Mission Control里设置，比如选择按照使用情况自动排列等。

54．隐藏程序

当我们不想在使用当前程序的时候看到其他程序的时候，可以使用快捷键option+command+h，这时除了你正在使用的程序，其他所有的程序都会被隐藏起来，有助于你专心工作。想切换到其他程序时，可以使用command+tab。

55．文件颜色标签的使用

OS X的Finder提供了颜色标签的功能，可以直接为文件和文件夹标

记颜色。我在很长一段时间都没有注意到这个功能，一次偶然的机会开始使用颜色标记文件，感觉非常方便。

比如我会在Finder的主目录下用颜色标明最常访问的文件夹。如果是电子书，可以用颜色表示阅读状态，例如绿色表示正在阅读，灰色表示读完了，橙色表示待阅读等等。大家可以根据自己的习惯使用颜色标签，提高效率。

56. 利用邮件中的日期创建日历事件

工作中我们总是通过邮件来通知会议和活动，这时邮件中往往有日期信息。我们可以利用这个信息直接创建日历事件。打开邮件，把鼠标移动到有效的日期信息上，会出现下拉菜单的按钮，点击后可以为日历添加事件，事件标题默认为邮件标题。

57. AppleScript小程序

今天为大家介绍用AppleScript实现一个示例小功能：清空废纸篓。打开AppleScript编辑器，输入如下代码：

```
——操作对象是Finder
tell application "Finder"
   ——为isEmpty变量赋值
   set isEmptyto "是否清空废纸篓！"
   ——显示确认对话框，点击确认程序继续执行，点击取消终止程序
   display dialog isEmpty
   ——清空废纸篓
   empty the trash
   ——通过语音说这事搞定了
   say "It is done!"
endtell
```

点击工具栏的编译按钮，检查没有错误后，点击运行即可，大家可以看看发生了什么。

58. Homebrew

Homebrew的功能和OS X自带的MacPorts很像，但是更为轻量级，由于大量利用了系统自带的库，安装方便，编译快速，实在是OS X系统开发中之必备工具。

安装方式：

```
ruby -e "$(curl -fsSL https://raw.github.com/mxcl/homebrew/go)"
```

使用方式：

```
brew install wget //安装wget工具。
```

具体的使用请参考：https://github.com/mxcl/homebrew/wiki

59. 根据文件名快速查找文件

我们在OS X中查找文件或文件内容一般使用spotlight或Alfred，这些功能以前的Mac Tips中都介绍过，不过，如果你知道文件名的一部分，想更加快速的定位文件，那么就会用到命令行工具locate。

locate是Unix/Linux下的命令工具，基本原理就是通过定期更新系统的文件和文件名并把索引信息放入系统的数据库中，当通过locate查找文件时直接从数据库里取数据。而且locate可以查到spotlight查不到的系统文件。

基本的使用方法非常简单，比如你想找niginx的配置文件在哪，只需输入：

locate nginx.conf

60. 设置用户登录选项

OS X系统登录后会自动启动一些程序，比如Alfred、拼音输入法、风扇控制软件等等，有时我们会嫌多，有时又想增加一些启动项，在哪设置呢？

打开系统偏好设置-用户与群组，选中当前用户，点击右边的登录项，你就会看到系统启动时加载的程序，可以随意删减，还能够设置启动后隐藏，非常方便。

61. 修改你的登录窗口

我们默认登录OS X时，系统会显示登录用户列表，你需要用鼠标点一下要使用的用户，或者用光标键选择用户，出现登录框后输入密码登录。如果我们想不显示用户列表，直接输入用户名密码登录怎么办呢？

打开系统偏好设置-用户与群组，点击左侧下方的登录选项（很奇怪很多人找不到这个），在右侧修改登录窗口为名称和密码。注销登

录，这次大家就满意了。

62．Mac的键盘

很多人第一次用Mac的键盘是会发现，苹果也太抠门了，退格键没了，PageUP/PageDown/Home/End也没了。别担心，您不是还有delete键和上下左右方向键么？delete相对于退格键，fn+delete可以往前删，fn+上下左右方向键可以实现PageUP/PageDown/Home/End的功能，一个都不能少。

63．QuickTime

很多人都会使用QuickTime Player看mp4或mov视频文件，但其功能远不止于此。option+command+n，可以打开录像功能，ctrl+option+command+n可以打开录音功能，ctrl+command+n可以打开录制屏幕功能，最后一个功能非常适合做产品介绍或产品演示，大家可以试一试。遇到快捷键冲突的，在QuickTime的文件菜单也可以找到这三项。

64．Dropbox快速导入Mac

有读者问如何把iPhone或iPad里的照片导入Mac，我自己用的办法是Dropbox，安装了Dropbox之后，每次用USB连接iPhone或iPad时，程序都会提示是否有新照片需要导入，导入后自动云端同步。不用Dropbox的同学，另外一个简单的方式是连接移动设备时，打开预览程序，点击文件，可以看到一个“从iPhone/iPad导入”的菜单，点击一下，后续你基本就知道该怎么做了。

当然还有其他方法，比如打开图像捕捉或iPhoto程序等。

65．快速创建日历事件

OS X提供了智能创建日历事件的功能。打开日历程序，点击左上角的+号，在弹出的输入框里输入：明天上午9点到13点参加公司年会。回车，看看效果如何？日历程序会准确的创建你想要的事件。大家可以试试其它写法。

66．显卡监控软件gfxCardStatus

现在大部分Mac都有两块显卡，集成显卡和独立显卡。OS X会根据

不同的程序自动切换显卡，但有时候我们在电池供电的情况下会由于某些程序的原因一直使用独立显卡，会大大缩短待机时间，这时候就能用到这个软件了。gfxCardStatus能做的事情有两件，一件是手动切换显卡。另一件是监控现在系统在使用哪块显卡，如果是独立显卡的话，是因为哪个程序导致必须使用独显。下载地址：http://gfx.io

67. 创建智能文件夹

Finder提供了智能文件夹的功能，简单来说就是固化你的搜索条件，并形成文件夹存放在左侧边栏。

例如你想建一个文件大小大于1G的智能文件夹，使用快捷键option+command+n呼出新建智能文件夹界面，点击最右侧的加号，在条件选择第一栏选择大小，第二栏选择大于，第三栏输入1G，你就可以看到你的Mac上文件大于1G的列表，点击存储，命名后该文件夹就会出现在左侧边栏。随时点击随时动态监控自己的硬盘上有哪些超过1G的大文件。试试其他搜索条件吧！

68. 自动打开程序文稿

OS X提供了自动恢复上次关闭程序时打开的文稿和窗口的功能。这就是说，如果你使用预览程序打开了5个PDF文件，用command+q关闭了预览程序，下次打开预览程序时，会自动恢复这5个PDF程序，包括文字选中的状态，阅读进度等信息。这个功能我非常喜欢，但有时候我们并不希望自动恢复，那么有两种方式可以关闭这个功能。

第一种：打开系统偏好设置-通用，选中“退出应用时关闭窗口”，这样所有的程序都不再具备恢复功能。

第二种：退出程序时使用option+command+q而不是command+q，相当于关闭所有文件并退出程序，下次打开时，这些文件就不会自动打开了。

69. 智能邮箱

邮箱账户的创建相信一般的用户都可以正常操作，不知道你是否使用过OS X中Mail的智能邮箱功能呢？

打开邮件程序，点击邮箱-新建智能邮箱，在弹出的窗口中选择你的过滤条件，过滤条件非常灵活，可以定义与或关系，增加多个过滤条件，设置完成后保存即可，你会发现左侧栏多了一个智能邮箱，点击即可根据你设置的过滤条件找到那些符合条件的邮件。

70．隐藏的VIP

如果你的系统是10.8以上，那么你就会发现邮件程序中多了一个隐藏的VIP功能。随便找封邮件，把鼠标放在发件人或收件人的邮件地址上，会出现一个蓝色的选择框，点击其中的白色箭头，在下拉菜单中点击“添加到VIP”，你就会发现左边栏多了一个VIP分栏，点击加入的VIP用户，可以直接查看他们发送的邮件。

71．在Finder中打开某个文件夹下所有子文件夹

有时候我们希望在Finder中查看某个文件夹下的所有文件和子文件夹，怎么做到呢？把文件切换到列表视图（command+2），把排序方式设置为不排序，这时文件夹左侧会出现一个箭头。按住option键点击文件夹左侧的箭头，你就会发现所有的文件和文件夹都展现在眼前了。注意，如果该文件夹下文件太多，不建议使用，打开会需要很长时间。

72．慢速动画

所有具备动画效果的操作，按住shift键，会播放慢速动画。大家可以试试按住shift键的时候最小化窗口，效果非常酷。

73．XtraFinder插件

这个插件具备和TotalFinder类似的功能，支持tab、文件夹置顶、多窗口、剪切、全局热键等功能，重要的是这是一个完全免费的自由软件。下载网址：http://www.trankynam.com/xtraFinder/

74．macbook待机，iPhone仍可供电

我们平时会把iPhone接到macbook上充电，事实上把macbook合上待机时，仍然可以为iPhone供电，大家可以试一下。如果你出游时会带上你的mac，别忘了这也是一块大的移动电池。

75．恢复截屏损坏图片

截屏图片存哪了？

OS X自带截屏不好使了，截屏之后有“咔嚓”的程序运行声，但图片不知道去哪里了，如何修复？

OS X自带的截图文件是存储在桌面上的，你的可能是被修改过了，我们可以通过以下命令恢复默认路径：

defaults delete com.apple.screencapture location

注销重新登录，再次截屏看看文件是否保存在桌面上了。

76．为OS X自带的字典增加中文词典

如何为OS X自带的字典增加中文词典？

目前OS X自带的字典程序是没有中文的，不过我们很容易为其扩展新字典。操作如下：

1、如果打开了字典程序，关闭。

2、到以下网址下载朗道英汉和汉英词典，解压缩得到两个后缀为dictionary的文件

3、把这两个文件复制到~/Library/Dictionaries下

4、启动字典程序，你就会看到增加了朗道英汉字典和朗道汉英字典

77．文件共享

在Mac之间进行文件共享有很多种方式，介绍两个最简单的，具备AirDrop功能的两台或多台Mac，在开着WI-FI的情况下打开AirDrop，就会找到同样打开AirDrop的Mac，把想传送的文件拖放到其他人的Mac头像上即可。

另一个就是利用系统的共享功能。打开系统偏好设置-共享，点击左侧栏的文件共享，在右侧区域配置即可。

78．删除程序

删除Mac上的程序有很多种，比如直接去应用程序文件夹下删除、用CleanApp删除等等，今天介绍一个最好玩的。

打开launchpad，按住option键，就会看到所有的程序图标都会像iOS图标那样晃动起来，点击图标左上角的叉，即可删除程序，操作和iOS一样。

79．command+上下方向键

这两个快捷键很多应用程序都支持，具体功能就是屏幕滚动到应用程序的顶部或底部，类似很多网站提供的“回到顶部/底部”功能。

Safari、Chrome、Firefox、Pages、Evernote等默认支持这样的功能。

在使用快捷键呼出Spotlight的时候，使用command+上下方向键还可以在搜索分组之间切换，非常方便。

80．Mac上的阅读笔记类软件

1、Kindle for Mac：支持视网膜屏，支持本地阅读和Amazon商店，支持中英文字典，电子阅读体验一流。遗憾的是不能整合中国和美国Amazon的帐户，导致电子书商品也没法使用同一个帐户阅读。（免费）

2、Caffeinated：优秀的RSS阅读器，如果你还喜欢博客和传统阅读，那么推荐使用。（收费）

3、Pocket：最好的稍后读App，支持标签分类、编辑等功能，支持Safari、Chrome等插件，非常适合知识积累（免费）

4、Evernote：很好的笔记类App，5.0之后UI有了很大的改进，目前我所有的文章都是用Evernote管理。（免费，有收费版本）

以上四个App在iPad、iPhone上也有相关应用，并且都支持云同步，合理使用对提高读写效率非常有帮助。

81．查看电源状况

按住option键，点击右上角的苹果－系统信息，在打开窗口的左侧栏中找到电源，点击即可查看电源的详细信息，主要的指标包括电池循环计数、状况等信息。如果您安装了Alfred，呼出后直接输入sys，也可以找到系统信息。

如果想简单查看一下电池的使用状况，按住option键点击顶部工具栏上的电池图标，可以显示电池使用状况。如果出现“尽快更换”、“修理电池”等信息，那么有可能是电池出了问题，建议先重置系统管理控制器（SMC），如何重置可以去Apple的官方支持网站查一下。还没效果的话，可能就需要换电池了。

82．Pixelmator

这款图像处理软件号称Mac上的精简版PhotoShop，而且更为人性化，适合非专业人士使用，不是平面设计人员也可以作出非常专业的图像设计。MacTalk里很多配图我都使用这款软件加工过，很好用。收费软件，但值得拥有。

推荐一个Podcast视频教程：http://www.pixelmator.com/tutorials/itunes/

83．搜索命令mdfind

mdfind是一个非常灵活的全局搜索命令，类似Spotlight的命令行模式，可以在任何目录执行文件名、文件内容进行检索，例如：

```
mdfind苹果操作系统
//搜索文件内容或文件名包含苹果操作系统的文件
mdfind -onlyin ~/Desktop苹果操作系统
//在桌面上搜索文件内容或文件名包含苹果操作系统的文件
mdfind -count -onlyin ~/Desktop苹果操作系统
//统计搜索到的结果
mdfind -name苹果操作系统
//搜索文件名包含苹果操作系统的文件
```

84．元信息命令mdls

mdls可以列出某个文件或文件夹的所有元数据信息，针对不同文件显示不同的元数据信息，例如文件创建时间、类型、大小等，如果是图片或音视频文件，则会显示更多元数据信息。使用方式非常简单：

```
mdls ~/Desktop/a.jpg
```

如果想查看图片的ISO数据，可以使用如下命令：

```
mdls ~/Desktop/a.jpg|grep ISO
```

85．功能键

很多程序猿在调试程序的时候总会用到f7、f8这些键，但在OS X里这些功能键默认分配了一些功能，想使用的话需要同时按fn+f8…

如果希望将这些f按键用作标准功能键而且不需要按fn，可以执行以下操作：

打开系统偏好设置-键盘，选中“将F1、F2等键用作标准功能键”，启用此选项时，顶部一行按键将用作标准功能键（F1-F12），而不执行音量控制等特殊功能。启用此选项后，若要使用这些按键的特殊功能，请按fn，比如请fn+f8来播放音乐。

86．查看文件信息的命令：file

file可以查看相关文件的类型和属性，相对于mdls，这个更亲民一些，基本用法：file xxx.png，大家感受一下。

87．如何配置多种网络环境

我自己无论在公司还是家里都是DHCP自动分配IP，所以不需要进行网络环境切换。但有些用户有时自动有时手动，需要多套网络配置方案，每次修改实在是太麻烦了。曾经有人问我Mac上是否有这样的第三方软件？我说没有，因为OS X的网络设置本身就提供了这样的功能。

打开系统偏好设置-网络，点击位置下拉菜单，找到编辑位置，打开后即可增删编辑多套网络设置，设置完成后保存。

这时点击屏幕左上角的苹果图标，在下拉菜单里增加了一个位置选项，里面就是你配置好的多种网络设置，点击切换即可。

88．生成man page的pdf文档

打开OS X的终端，通过man命令可以直接查看该命令的使用手册，但有时我们会觉得在命令行查看不太方便，如果可以提供一个pdf文档就完美了。这很容易做到，在终端输入如下命令，即可在预览程序打开grep的使用手册，另存为你需要的文件名即可：

```
man -t grep |open -f -a Preview
```

89．虚拟机

2006年Mac的硬件进行了重大的架构调整，开始全面采用Intel系列CPU，Power渐行渐远。架构的调整和Bootcamp的推出，使得在Mac上安装双系统变得触手可及。基于Mac的虚拟机应用也开始出现。我刚开始使用Mac时是双系统的支持者，后来Windows用的越来越少，就比较推荐使用虚拟机了。

在OS X上主要有三款虚拟机软件：Parallels Desktop, Vmware Fusion和VirtualBox。简单给大家介绍下：

·Parallels Desktop：Parallels是OS X上一款优秀的虚拟机软件，最新版本是8。它支持多种操作系统，并对Windows有完美的支持。通过融合模式，可以让Windows程序运行起来象Mac的应用。提供把Vmware Fusion虚拟机迁移到PD上的功能。收费。

·Vmware Fusion：Vmware在Windows和Linux下大名鼎鼎，Fusion是Mac版本，功能同样强大。收费。

·VirtualBox：Sun推出的一款开源虚拟机，现在归Oracle了，未来走势不明。免费。

我个人首推Parallels Desktop，功能、性能和价格都不错，专注于桌面版，属上乘之选。我自己虚拟了Win7、Reahat Linux和Ubuntu等环境，作软件测试和搭建多机开发环境。

90．如何开启root用户？

用过Linux/Unix系统的都知道root用户，它具备读写文件系统所有区域的特权，是最高级别的用户。OS X一样有root用户，只不过默认情况是不开启的。我们想在命令行执行需要root权限的操作时，可以在命令之前增加sudo指令，比如执行每日维护指令，sudo periodic daily，系统会提示你输入用户密码，执行root权限。在GUI（图形界面）执行root级别的命令时也会提示输入用户密码。一般情况下我们是不需要开启root用户的。

用惯了Linux系统的用户有时很想启用root用户，其实也很简单，打开Finder，输入shift+command+g，在前往文件夹中输入：

/System/Library/CoreServices

然后在目录中找到目录实用工具并打开，解开左下角的小锁，然后点击顶部菜单的，你就会看到启用或停用root用户的选项了。然后我们在命令行下执行su -，就可以切换到root目录下，root的默认目录是/var/root。

root有风险，启用须谨慎！

91．隐藏的空间切换功能

以前介绍过OS X中Space的使用，我们可以定义多个Space，每个程序都可以在特定的Space中打开，多手势上推下滑选择程序，也可以通过ctrl+数字切换Space，很方便。今天再为大家介绍一个隐藏的功能，就是通过四指双击触控板，可以在你最近使用的两个Space之间切换，这个功能就类似电视频道中的返回功能，当你使用了Space1中的一些APP，切换到Space4，通过四指双击可以在Space1和Space4之间切换，对于协同工作非常有效。典型的应用场景：在Space1里编码，在Space4里参考各类文档。

功能开启，打开终端程序，输入：

```
defaults write com.apple.dock double-tap-jump-back -bool TRUE;#功能开启
killall Dock;#重启Dock
```

92．免费的文本编辑器Imagine

我个人觉得Imagine比OS X自带的TextEdit好，除了目前不支持iCloud外，基本涵盖了TE的功能，而且排版简约美观，可更换柔和的背景色，全屏写字非常舒服，对字体样式的支持很好，在富文本和纯文本间切换方便，我基本用Imagine替代了TextEdit。下载地址：https://itunes.apple.com/cn/app/imagine/id566877440?mt=12

93．去除右键菜单的重复项

OS X系统有个问题，某个程序反复安装后，选中某种类型的文件，点右键-打开方式，你会看到不少重复的选项，我们可以用以下命令去除重复项。

```
/System/Library/Frameworks/CoreServices.framework/Versions/A/Frameworks/LaunchServices.framework/Versions/A/Support/lsregister -kill -r -domain local -domain system -domain user
```

94．如何分别设置Mac的鼠标和触控板的滚动方向

很多人习惯鼠标使用相反的滚动方向，而触控板类似iPad那样的自然滚动，问如何设置，当时我的回答是不知道，因为目前OS X的系统设置里，鼠标和触控板的设置是统一的。今天发现了一个免费的软件Scroll Reverser，可以实现鼠标和触控板的分别设置。下载地址：http://www.macupdate.com/app/mac/37872/scroll-reverser

启动后程序显示在顶部菜单栏，设置简单明了，有需要的用户体验一下吧。

95．如何让不支持Retina的Mac软件变成Retina App？

前两天有读者求推荐Mac下的FTP软件，我推荐了FileZilla，但这个软件是不支持Retina屏的，Retina用户使用这个软件会感觉整个世界都模糊了，结果搜索之下，发现了一个小软件，叫做Retinizer，顾名思义，就是把非Retina的软件Retina化，我用了一下，完美支持FileZilla。下载地址如下：http://retinizer.mikelpr.com/

96．文件比较

1、对于单个文件的比较，一般使用diff或vimdiff就可以了，比如：

vimdiff destfile.txt sourcefile.txt

vim会非常清晰的显示出文件的不同，还有很多快捷方式帮助你查看和操作文件，这个命令比较适合命令行爱好者。

2、对于大批量文件的比较，还是图形化比较工具更合适一些。OS X自带了FileMerge比较工具，可以满足部分需求，但对于中文编码文件或大文件经常会崩溃，很奇怪Apple一直不解决这个问题。

3、推荐一款收费软件，VisualDiffer（25元），UI、功能和稳定性都非常不错，实在是居家旅行、代码比较、查找问题的必备利器。

97．FTP工具Cyberduck

之前介绍Retinizer（普通软件Retina化）的时候提到了FTP软件FileZilla，我个人一般使用命令行下的ftp/sftp/scp等实现FTP软件的功能，但普通用户还是用图形界面的更方便些。今天再给大家介绍一个可以实现远程同步文件的FTP工具：Cyberduck。

Cyberduck除了可以实现FTP的基本功能外，还能支持远程同步。所谓同步，就是把远程和本地的两个目录进行比较，然后自动找出修改的文件上传到服务器。

具体操作就是通过ftp或sftp的方式登入远端服务器，选中某个文件夹，右键菜单里选择同步，再选择本地文件夹，就可以进行同步比较上转了，上传之前你最好确认下，更稳妥。

同样，这个软件也可以用Retinizer实现高清显示效果。

98．文件重命名

文件重命名的问题以前说过，但最近又有些用户问起，就再说一下。

如果你没有装任何插件的话，在Finder中重命名文件或文件夹的快捷键就是回车。打开文件用command+o，返回上级目录用command+向上的方向键。

如果你装了原来推荐过的XtraFinder，可以把回车改为打开文件（与windows操作类似），把option+r设置为文件重命名。

如果你在命令行下重命名文件，命令是这样的：

```
mv oldname newname
```

99．多个用户登陆一个程序

Mac下有很多程序默认是单进程的，比如你不能打开多个邮件程序，不能打开多个Evernote，但有时我们可能会有这样的需求，那么用如下命令可以实现：

```
open -n /Applications/XXX.app
```

-n的含义是Open a new instance of the application(s)even if one is already running，意思就是为正在运行的应用程序再开一个新实例。常用于多个账户登录一个程序，或软件比较等场景。

100．强制关闭程序

总有程序关闭不了，这时候我们就需要：

方法一：option+command+esc，调出强制退出应用程序的窗口，选择要退出的进程即可。

方法二：打开活动监视器，类似windows的任务管理器一样操作就好了。

方法三：命令行下的kill命令，比如想杀掉TextMate，首先用ps -ax|grep TextMate找到进程号，然后用kill -9进程号，即可。

至此，天下无杀不掉的进程。

101．用AppleScript实现打开多实例程序。

之前介绍了通过open -n /Applications/XXX.app的方式打开多实例程序，有人在微博上问如何选中一个文件或程序，通过右键菜单打开新实例，而不是每次都去命令行操作。

我们可以通过Automator+Applescript实现这个功能。

打开Automator，选择创建服务，在左侧选择“运行AppleScript”，双击打开程序窗口，在（*Your script goes here *）处输入如下代码：

```
tell application "Finder"
  try
    set filename to POSIX path of (selection as text)
    set fileType to (do shell script "file -b " & filename)
    if (fileType does not end with "directory")or (filename end with "App" )then
      do shell script "open -n " & filename
    end if
  end try
```

```
end tell
```

在程序上方的选择框设定“文件和文件夹”、“任何应用程序”，然后保存，起个你喜欢的名字，比如叫“以新实例运行”。退出Automator。

选中文件或程序，右键-服务-以新实例运行，即可实现类似open -n的方式。

102．Automator

Automator是苹果公司为其操作系统OS X开发的一款软件。通过点击拖拽鼠标等操作就可以将一系列动作组合成一个工作流，从而帮助你自动完成一些复杂的重复工作。Automator还能横跨很多不同种类的程序，包括：查找器、Safari网络浏览器、iCal、地址簿或者其他的一些程序。在Automator中可以运行Applescript。

在上一个技巧中我们通过Automator创建了一个服务，当你在Finder或桌面上选中文件时，在右键的服务菜单里增加了一个选项：以新实例运行，是通过Applescript实现的，下面说明一下程序功能：

```
——通知Finder
tell application "Finder"
  ——异常处理
  try
    ——获取选中文件的全路径
    set filename to POSIX path of (selection as text)
    ——通过脚本file -b获取文件类型
    set fileType to (do shell script "file -b " & filename)
    ——如果不是文件夹或以app结尾，执行open -n脚本
    if (fileType does not end with "directory")or (filename ends with "app")then
      do shell script "open -n " & filename
    end if
  end try
end tell
```

这里考虑到了选中程序直接打开，或选中文件以默认程序打开的情况。

103．Safari默认查询引擎查询应用软件文字

如果你想通过Safari的默认查询引擎查询某个应用软件里的文字，选中文字，然后shift+command+l，即可跳转到Safari的搜索页面，非常方便。大部分应用都支持这个快捷键。

104．旋转屏幕

打开系统选项设置，已经打开了的，退出重新打开。按住option+command键盘，点击显示器，在原来的亮度选项下方会出现一个旋转的选项，这时候你就可以旋转你的屏幕了。

105．keycastr录制视频屏幕上显示键盘快捷键

最近尝试录制视频时在屏幕上显示键盘快捷键的操作，ScreenFlow固然可以实现这个功能，不过99美金的价格让人感觉得不偿失。搜索之下找到了keycastr，简单设置了一下发现可以实现我需要的功能，项目托管在github上，可以直接下载dmg包。下载地址：https://github.com/sdeken/keycastr

还有一种方案是使用OS X原生的键盘显示，打开语言与文本偏好设置-输入源，选中左边栏的第一项：键盘与字符显示程序。关闭偏好设置，这时点击顶部menu bar的语言，会多出两项功能，点击键盘显示程序，就会在屏幕上出现一个模拟键盘。

这个方案的缺点是没法区分快捷键和普通字符输入，而且显示速度太快，不够醒目。

106．复制截屏图片到剪贴板

以前介绍过如何通过苹果自带的快捷键截屏并存储图片文件，例如shift+command+3和shift+command+4，现在发现如果在以上两个截屏动作中加入ctrl键，可以实现直接把图片保存在剪贴板而不是实体文件，这样你可以通过command+v直接把截取的图片内容复制到图像处理软件或Pages、Keynote等文件中。

107．CheatSheet

一生要记住多少快捷键呢？我都不知道自己记住了多少快捷键，很多快捷键是到了那个环境下才能想起来。但是毋庸置疑，快捷键可以大大提高我们的工作效率，在Mac环境下使用快捷键和不使用，几乎是两种体验。如何记住这些快捷键呢，有人开发了一款软件叫做CheatSheet，安装并打开之后，当你记不住快捷键的时候，按住command键两秒钟，就会弹出一个当前应用软件快捷键列表，不全，但是对大部分用户都够用了。下载地址：https://itunes.apple.com/cn/app/cheatsheet/id529456740?mt=12

108．HTML5Player

现在越来越多的人开始看在线视频，目前大部分视频网站的播放器都是基于Flash技术，而苹果一直对Flash很抵触，支持的也不好，Flash播一会Mac机身就会变热。另外现在的视频网站广告太多，页面花里胡哨也不适合观看。于是有位无聊的程序猿做了一个HTML5播放器，可以把在线视频的播放转化成HTML5方式，并且去除广告。使用起来非常简单，只要把{原文}里的HTML5Player链接拖拽到Safari的书签栏，播放视频时点击书签栏上的HTML5Player书签，播放器就会自动转换，效果自己看吧。

目前支持优酷，土豆，搜狐视频，爱奇艺，乐视网，QQ，迅雷离线，56视频的单视频播放页面。相关链接：http://zythum.sinaapp.com/youkuhtml5playerbookmark/

109．重建Spotlight索引

以前给大家介绍过，在OS X中几乎不需要进行文档和文件夹管理，因为有Spotlight机制，可以瞬间找到你想要的文件，只要你记得这个文件的一点蛛丝马迹。

但是Spotlight也有出问题的时候，就是它的索引文件出事了，比如查找速度变慢，某些文件明明在硬盘上就是检索不到，等等，这时候就需要重建索引了。

打开终端程序，输入如下命令：

```
sudo mdutil -i off /
#该命令用来关闭索引
sudo mdutil -E /`
#该命令用来删除索引
sudo mdutil -i on /`
#该命令用来重建索引
```

然后用快捷键呼出spotlight菜单，随便输入一个词，就能看到提示，正在进行索引，并且显示完成重建索引需要的时间。

完成之后，spotlight又可以运转如飞了。

有时候人在某个阶段也需要重建索引，保持初心。什么是初心，空空如也！不要成天得瑟你知道的那点事，多琢磨那些你还不知道的事儿。

110．用键盘操作Dock和menu bar的菜单

当我们想操作Dock或顶部菜单栏的时候，往往需要鼠标去选中Dock或菜单栏，但是我们往往是不希望去碰鼠标的，这时候快捷键就又开始发挥作用了。使用control+F2可以选中menu bar的菜单，通过左右键选择功能，回车执行；使用control+F3可以选中并显示Dock，通过左右键选择功能，回车执行。

该功能在全屏操作时尤其有效。对于F1、F2等不是标准功能键的设置，增加fn键即可。

111．定义自己的快捷键

我认为OS X是一个把GUI（图形界面）、程序进程、脚本结合的最好的操作系统，当然这样说可能有些读者不是很明白，这么说吧，OS X是一个定制化非常强的系统，很多人说OS X封闭，事实上OS X为用户预留了非常多的入口和切面，让你能够通过简单、简洁的办法进入系统做你想做的事情。

举例来说，对于普通用户，你可以通过键盘的快捷键设置定义自己的常用操作。

对于程序员，你可以自己通过AppleScript / Shell / Automator等创建自己的服务，也可通过类似Alfred 2这样的优秀工具编写自己的workflow。

今天给大家说说第一种，打开系统偏好设置－键盘－键盘快捷键，左侧栏里列出了各种功能的快捷键，比如Launchpad和Dock，Mission Control，截屏，服务等等，大家可以在这些选项种定义和修改自己常用的快捷键，增加右键菜单等等。

112．选择文本

用command+鼠标，可以选中不同位置的文本内容。

用option+鼠标，可以对文本进行块选。

113．Dock中的文件夹

这个功能非常适合普通用户使用。一般安装了系统后Dock右边会有几个默认的文件夹，事实上你可以把任何常用的文件夹拖到这个位置，不想要的拖到废纸篓即可移除。

Dock文件夹的显示方式提供了扇状、网格和列表三种方式，我一般使用网格和列表，但是还有一个隐藏的列表功能，更为实用些，可以在

命令行输入如下命令开启：

```
defaults write com.apple.dock use-new-list-stack -bool TRUE; killall Dock
```

这时候你再启动列表模式，就会发现列表显示方式不一样的，变得更加容易操作。

另外，在列表和网格模式，还可以通过command +/-来放大和缩小图标，非常方便。

114．Finder的宽度

Finder是OS X的默认文件管理器，它提供了多种显示方式，包括图标、列表、分栏和Cover Flow。其中分栏最为常用，通过键盘的方向键浏览多层级的文件非常方便。不过每个分栏的宽度都是系统默认宽度，如何改变这个默认宽度呢？用鼠标拖动分栏线时同时按住option键，这个默认宽度就随之改变了。

115．Dashboard

顾名思义，Dashboard就是OS X系统中的仪表盘，它可以在桌面上显示各种小功能块，比如字典、便签、系统状态、天气预报等。

使用快捷键f12或点击Dock中的Dashboard可以运行Dashboard，运行方式可以在一个新的Space里，也可以在当前的Space里，设置在偏好设置-Mission Control中。我一般使用在当前Space里打开。

点击左下角的+号，可以为Dashboard添加功能块，-号可以删除已经添加的功能块。把鼠标移动到某个功能块时按住option键，该功能块会出现一个删除图标，点击也可删除。

如果你想添加更多的功能块，在点击+号时，右侧会显示更多Widget，点击可以到网络上下载你需要的功能。

Dashboard还有一个Web Clip的功能，如果你添加了这个功能块，浏览网页看到特别喜欢的词句或图片，可以点右键-在Dashboard打开，把这部分内容放入Dashboard。

116．Dock文件夹的使用小技巧

有一期介绍过Dock文件夹的使用问题，再说一个小技巧，当我们打开Dock文件夹后，先打开某个文件所在文件夹时，按住command，点击该文件，就会打开Finder文件夹，并选中你刚才点击的文件。

117．介绍几个简单的命令

打开终端程序，输入date会显示当前日期，输入cal会显示日历，输入uptime会显示系统从开机到现在所运行的时间。

118．神奇的option键

以前很多期介绍过option相关的快捷键和功能，比如选中多个文件option+右键，可以显示检查器，按住option点击顶部菜单的电池会显示电池状况，点击wifi会显示网络状况，点击备份...可以点点试试。别忘了最左边的苹果按钮，option+点击，在下拉菜单点关机、重启都不提示的。

option+拖拽文件可以复制，按住option输入=输出是≠，按住option和shift输入=，输出是±。

还有好多，没事的时候多按按option键，你会有很多意外的发现。

119．音乐处理软件XLD

XLD全程是X Lossless Decoder，是Mac平台上无损音乐播放、编码和转换工具，不仅支持APE、FLAC等无损音频，还支持读取音频CD，将音轨抓取出来之后创建音乐文件。

免费软件，喜欢的可以捐赠。官网地址：http://tmkk.undo.jp/xld/index_e.html

120．保护你的数据文件

在Mac下对某些文件或数据进行加密操作有两种方式：

第一种：系统偏好设置-安全性和隐私-FileVault，打开FileVault即可。FileVault是全盘加密技术，可以对磁盘上的所有文件进行加密，后果是系统速度会稍微变慢一点点，如果你不是在军方服役，一般不建议采用。

第二种：创建磁盘映像文件，对磁盘映像进行加密处理，然后把需要保护的数据和文件放到这个磁盘映像中即可。具体方式如下：

打开应用程序-实用工具-磁盘工具，点击新建映像，在加密选项处选择256位AES加密，这种加密算法是极其安全的。创建映像时输入两次密码，即可创建加密的磁盘映像文件。在创建时最好不要选择“在我的钥匙串中记住密码”，这样可以每次打开这个磁盘映像文件时都需要输入密码，可以达到最佳保护数据的作用。

121．如何禁用通知？

很多时候写作或写代码，不希望被打扰，这时候就需要把OX S的通知关掉，双指从触控板右侧滑入，呼出通知中心，在最顶部有一个显示提示和横幅的开关，关掉就会禁止通知，不过第二天会自动回复这个通知设置。

更简单的做法是按住option键点击屏幕右上角的通知图标。

122．Finder的工具栏

我们可以把文件和程序拖到Finder的工具栏上，以便随时打开。但是想移除时会发现点击鼠标拖动是没法把这些图标移除的，这时候只要在点击拖动时加上command，你就会发现这些图标被销毁了。

123．Spotlight搜索command*键定位

用Spotlight搜索的时候，搜到文件时，我们有时候会需要打开该文件所在的文件夹，这时候按住command*键，点击文件即可打开Finder，并定位到该文件所在文件夹。

124．重新启动Finder快捷方式

Finder是OS X系统中的常驻程序，一般不需要退出，如果想重新启动Finder时，有一个简单的方式，按住option键，右键点击Dock上的Finder图标，底部菜单会出现重新开启的选项，点击即可。同样的操作对其他Dock上的程序是强制退出。

125．屏幕画中画

之前介绍过屏幕放大功能，也就是通过option command ±可以放大和缩小屏幕，使用control +滚轮也可以。

不过这只是放大屏幕方式的一种表现形式，我们还可以通过辅助设置改为画中画模式，打开系统偏好设置-缩放-缩放样式，把全屏幕改为画中画即可，效果大家自己看吧。

126．粘贴纯文本

我们在网页或其他文档上复制文字的时候，会把文字格式一并复制下来，command+v会把文字格式都粘贴过去，如果我们只想粘贴纯文本，可以使用shift+option+command+v，大部分软件都支持这种方式复

制纯文本。

127．终端命令lsof

有用户问，在倾倒废纸篓的时候，经常会提示该文件还在使用，不能删除，但是又不知道哪个程序在用，怎么办？

Unix下有一个命令叫做lsof，名字是list open files的缩写，顾名思义，就是查看打开的文件，在终端里输入lsof文件名，就可以找到打开这个文件的程序。关掉程序，就可以正常删除文件了。当然lsof还有很多丰富的指令，感兴趣的用户自行Google吧。

128．AirDrop的有线传输

Airdrop默认只能通过WIFI来传文件，如果电脑已经连了网线，但是没开WIFI就不能用AirDrop了，有一个办法可以打开AirDrop通过有线传文件的特征。打开终端输入：

```
defaults write com.apple.NetworkBrowser BrowseAllInterfaces 1
```

然后选中Dock栏的Finder，按住option键右键点击Finder图标，点击底部菜单项“重新开启”，Finder重启之后，即使你的电脑没开WIFI，也可以用AirDrop给别人分享传文件了。

129．切换程序时实现预览功能：

通过command+tab可以实现程序之间的切换，如果我们想在切换到某个程序的时候看看该程序组都在显示什么，可以按住command的同时按数字键1或上下方向键，系统会调出该程序的Exposé模式，这时你可以放开所有按键，用鼠标或方向键选择显示哪个程序窗口。

130．Spotlight检索的高级技巧

·通过文件类型搜索文件，搜索格式是：
kind:文件类型搜索关键字，比如：

kind:app——搜索应用程序
kind:bookmark——搜索书签和历史纪录
kind:contact——搜索联系人
kind:document——搜索各类文档
kind:word——搜索word
kind:pages——搜索pages

kind:key——搜索keynote
kind:email——搜索email
kind:event——搜索日历事件
kind:folder——搜索文件夹
kind:movies——搜索视频
kind:music——搜索音乐
kind:pdf——搜索pdf文件
kind:pic——搜索图片
……
·通过标签颜色搜索
如果你喜欢使用各种颜色的标签标注不同的文件夹，那么这个功能就用的上了。
label:红，就可以找到红色标签的文件和文件夹。
·通过日期搜索
date:today——查看今天创建或修改的文件
date:yesterday——查看今天创建或修改的文件
date:2013-05-01——查看2013年5月1日创建或修改的文件
·条件表达式
想搜索包含Mac不包含Windows的Keynote，可以这样写：

kind:key Mac -Windows

也可以这样写：

kind:key Mac NOT Windows

我们可以使用+/-进行条件表达式求值，也可以通过NOT AND OR来检索，不过后者一定要大写，否则会被当做搜索内容处理。

有了以上4种搜索方式，天下再无搜不到的文档！

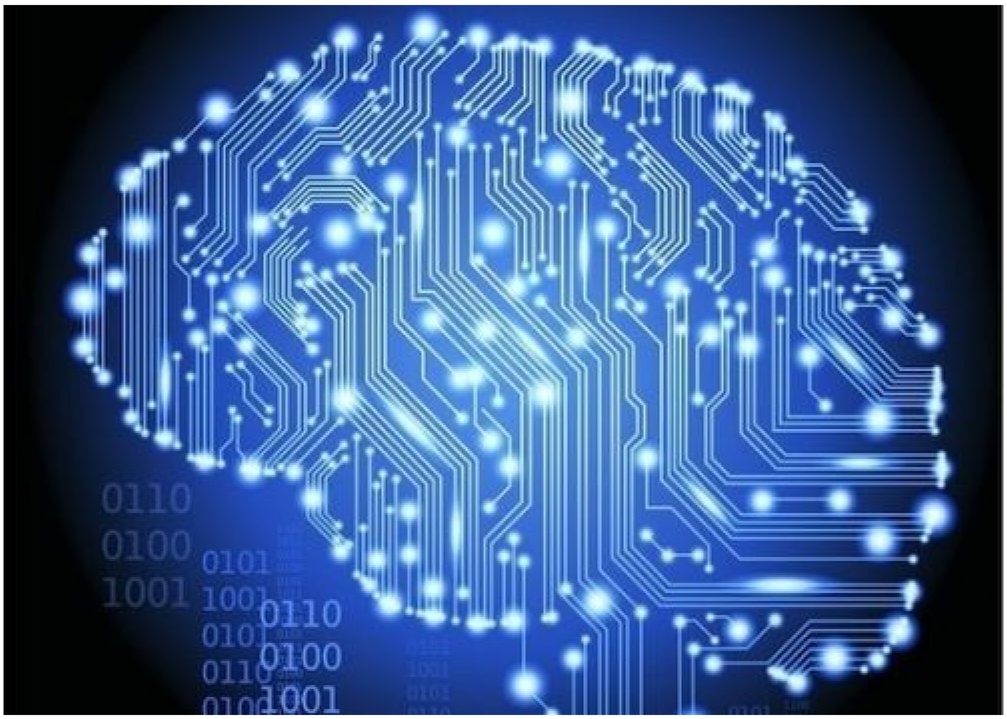

MacTalk

程序员与编程

并发的错觉

【发布日期　2013年7月1日】

今天聊一聊电脑和人脑的并发问题。



在计算机发展初期，CPU的计算能力非常有限，计算资源稀缺而昂贵。最早的时候一个CPU只能同时运行一个任务，这简直让人无法忍受。什么叫做只能运行一个程序呢？这就像大学上自习占座一样，一旦一本书、一张纸、一个包或一个活人占有了那个桌子，其他人就再也没法用了，无论是这个人出去上厕所，踢球，你都不能去用那个座位，如果你贼胆包天敢偷着去坐，这时候就会有个神秘人突然拍拍你的肩膀告诉你“童靴，这里有人”，这就是常说的“见鬼的故事”。故事里的座位就是CPU，无论当前任务在使用CPU进行计算，还是在读写磁盘IO或进行网络交互，丫都得占着CPU，黑客极客和各种无证程序员们觉得，这，不，科，学！

于是大家试图通过各种方式来改变这一现象。首先出场的是多通道

程序，程序员们很快写了一个监控程序，发现当前任务不用CPU计算时，就唤醒其他等待CPU资源的程序，让CPU资源能够得到充分利用。但多通道的问题是调度乏力，不分青红皂白和轻重缓急，不管是急诊还是普通门诊，该等都得等。

第二出场的是分时系统，分时系统是一种协作模式，每个程序运行一小段时间都得主动把CPU让出来给其他程序，这样每个程序都有机会用到CPU一小段时间。这时操作系统的监控程序也完善了一些，能够处理相对复杂的请求。早期的Windows和Mac OS（注意没有X）都是采用这种方式来调度程序的。分时系统的问题是，一旦某个程序死循环，系统就没招了，只能干等着，就像死机了一模一样，程序员们说，这是不可接受的！

第三个隆重登场的是多任务系统，程序员们让操作系统接管了所有的硬件资源，变得更加高级智能，系统进程开始分级，有的是特权级别，有的是平民级别（你就知道，在计算机世界都特么是这个样子！），所有的应用程序以进程和线程方式运行，CPU的分配方式采用了抢占式，就是说操作系统可以强制把CPU的资源分配给目前最需要的程序。程序员们成功了，几乎完美的控制了一切，并造成了很多任务都在同时运行的假象，如果用两个字来形容的话，那就是“和谐”！目前OS X、Unix、Linux、Windows都是采用这种方式进行任务管理的。

以上都是单核单CPU的情况，但无论线程间的切换多么快，这些都是并发，而不是并行。

好吧，中间插播一段并发和并行的区别。并发的英文单词是Concurrency，并行是Parallelism。如果一个系统支持两个或多个动作（Action）同时存在，那就是一个并发系统。如果一个系统支持两个或多个动作同时执行，那就是一个并行系统。也就是说，单个CPU永远无法同时执行两个或以上的任务，但是允许任务同时存在。所以，只有多核或多个CPU才可能发生并行，如果单核单CPU只能发生并发行为。

如果有人以为单核单CPU的并发就是同时执行很多任务，那么这是个错觉。

不知道解释清楚了木有。插播完毕！

终于，多核CPU和分布式系统被干出来了，一台计算机可以拥有多颗CPU，每颗CPU可以有多核，同时，成千上万台的机器被连接在一起进行计算，大家一看都晕了，史称“云计算”。随着硬件的变化，软件技术同时开始革新，各种语言开始支持并行计算，比如Erlang/Scala的Actor&Message模型，Go语言的goroutine机制，Java的ForkJoinPool，Objective-C的Grand Central Dispatch技术，当然还有Hadoop等分布式框

架。

总之，到了目前这个阶段，无论是并发，还是并行，计算机和CPU都算是解放了，它们不仅在单台机器上可以执行并行计算，在横向扩展上也变得随心所欲，各种云平台应运而生，公有云私有云混合云balabala，反正是比较晕……

人脑就比较惨了，在电脑突飞猛进的这几十年里，几乎没有任何进展，脑袋仍然只有一个，也没有裂变出多核……

上文书谈了电脑的并发和并行的事情，有读者反馈，像一个有趣的教科书！我特么最烦教科书，就因为教科书，哥写了几年程序才把这些事捋清楚，为了让你们不再重蹈覆辙，我容易吗我！

好，下面我们谈一下人脑的并发，先看一个读者反馈，你们感受一下：

> 看过一个关于人脑的理论，说不清是并发还是异步。有时候我们很努力的想一个问题，但却怎么也想不起来，于是我们放弃了。但是大脑并没有放弃，此时它会自动起一个gorouting，继续再大脑的各个角落去寻找这个记忆碎片，当找到时执行回调告诉你。而此时你可能正边洗澡边哼着小曲儿！

关于人脑的机制，其复杂程度超过CPU何止万倍，比如上古奇人周伯通郭靖小龙女的双手互搏到底是并发还是并行呢，Mac君万不敢断言，未来还是让研究人脑图谱的人去探索其真正的奥秘吧。我们在这篇文章中把“一心二用”或“一心多用”统称为人脑的并发。

俗话说“一心不能二用”，这句话常常送给那些做事三心二意的人，但是我们真的不能一心多用吗？或者说并发带给我们的到底是效率的提升还是状态的下降？关于这件事Mac君的看法是，不可一概而论。“好吧，那位同学请把砖头继续放入怀中，我们还没有讲完”。

关于人脑的多任务处理，应该从个人特点、所处环境和任务特性来考虑。

其实人脑天生就是用来处理多任务的，比如你可以一边洗澡一边唱歌，一边看电影一边磕瓜子还要注意不要被飞来的砖头砸到等等，不过这样的多任务都是在放松环境下的简单任务，对我们提升效率没什么意义。

但是，当我们在健身房跑步时听英语，写文章或编码的时候听歌（所有不让听音乐编程的公司都将死于心碎），坐地铁的时候阅读，步行的时候思考，这就变得非常有意义的，因为我们在一个相对宽松的环境下把复杂的逻辑任务和简单的机械任务结合在一起，既不影响A，也不会干扰B，这种情况是我们优先要采取的并发策略。

类似的事情，比如开车时听英语，就要因人而异了。我有近10年的驾驶经验，喜欢开车，驾驶基本上已经形成下意识的动作，从出发到目的地往往不会记得自己做了哪些操作，所以我经常开车时听英语并有所收获。但有些人开车仅仅是驾驶已经够紧张忙乱了，倒一次车能车头入绝不车尾进，开次长途出的汗够洗澡的，那么就专心开车好了，车内最好保持安静或听一些舒缓的音乐。

我曾经看过一本叫做《错觉》的书，书中有一段描述了一位机长在飞机飞行的过程中发现机上设备出了点小故障，于是他和副机长一起排查，接着又找来机械师，哥三忙的不亦乐乎，过了一段时间，有人问，谁在开飞机呢？这时飞机无人驾驶已经很久了，等反应过来之后，飞机已经开始俯冲坠地，机上人员全部罹难！这种空难并不是意外，一架状况良好的飞机直接撞向地面不是偶尔发生，这种现象在航空领域被称作“可控飞行撞地”，其根本原因就是，人们太相信自己的多任务处理能力！

驾车虽然比驾驶飞机简单多了，但同样是一项非常危险的工作，所以我建议大家，听听音乐就好，另外千万别玩手机。

还有一种情况就是，在同一时间做两项或多项复杂任务，比如你让程序员在编码的同时帮助别人解决问题，能不能做好？也许有人可以，但我的感觉是，这种安排效率反而会打折扣。人们在很多时候会低估自己的能力，但在更多时候会高估自己。在复杂任务并发处理的时候，人脑往往会高估自己的处理能力，以为可以，其实任务的并行，上下文的切换，注意力的分散，都会让你的效率大打折扣，所以设计模式中的职责单一原则不是盖的，一个类尽可能只做一件事情，无论是效率还是后期维护都会好很多，人脑其实也是一样。

总结一下：

1.简单任务的并发是大脑天生的nature，每个人都在不自觉的应用。

2.在宽松的环境中让简单机械的任务和复杂有机的任务并行完成是非常不错的做法，提高效率节省时间。

3.在高危环境中（驾驶、高空作业等等）我们应该专心致志的只做当前的工作。

4.对于复杂任务，我们最好一件一件完成，即使有些人能够同时处理多重任务，那也需要长期的艰苦训练，比如郭靖君，你能否做到，就得看有没有周伯通那样的大哥！

程序员的性格

【发布日期　2013年6月6日】

今天的话题来自一封读者来信，他在邮件里写了大概2000多字，其中一大部分描述了自己的经历，我觉得非常精彩，写过汇编写过C，玩过Flex，做过web（HTML5+CSS+JS），还用过Scala和OC，经历过创业人生，整个技术历程也足够丰富，可能比很多技术人员丰富。这样一个典型的程序员遇到的问题是什么呢？如下：

> 我是个静下来喜欢反思自己的人。我虽然对自己过去的一年挺自豪的，但我也发现了在我身上发生了一些变化。我发现我的幽默感渐渐消失，而且我很喜欢从“另一面去看问题”，一开始我觉得这是我具备了独立思考的能力，不会轻易被别人的言论左右。但这也给我身边的人带来了困扰。因为当他们说出一些东西的时候，我首先的反应就是：可能并不是这样，那样也许更好，其实是这样的……等等，而且几乎是反射性的，不假思索就能找到问题的另一面，并且表达出来。然而说者无意，听者有心。朋友之间，说说笑笑还好，他们最多觉得是我很喜欢抬杠、反驳别人，送我外号“杠王”。但我老婆和我生活在一起，就受不了了，觉得我太自负，自以为是，愤青。我自己知道，我不是故意这样的，但这已经成为了下意识的行为。虽然这现象以前就有，但是这一年变化的很明显，明显到我自己都察觉到了。我不想变成一个让身边人觉得不舒服的人，我也怕以后会发展成“阴谋论”者。我不是不懂得谦逊，只是对自己的观点比较执着，而且想强烈的表达出来，并且希望别人也赞同。到后来，我自己都分不清楚，我到底是真的发现了这些事物的“猫腻”，还是我为了反驳而反驳的。我觉得这是一个很不好的习惯，因为我身边的人和我探讨某些问题的时候，在我这里得到的几乎都是否定的结论。
>
> 就我关注的一些技术前辈，包括我同学，也都是这样。我发现我们这些技术同类，好像“戾气”都比较重。而您在我关注的人里，是属于比较温和的，没有那么张扬，没有显示出“自负”和“自以为是”，我想请教下，您是否遇到我这样的问题？而您又是怎么控制自己的？为了表述清楚，我可能把我的问题夸大了一点点，但我觉得如果我不改善的话，迟早会变成那样，甚至更坏。

其是我想说的是，大部分牛人都很张扬，“戾气”也重，我之所以比较温和，一个是岁数大了，另一个就是确实不够牛。另外的一点就是，当牛人牛到一定的境界，可能就会重归和平，比较文艺的描述就是，“大牛领会了返璞归真和万物生长的道理，知行合一，遇事抖抖衣袖，不溅起一片涟漪”。你现在浑身都是杠头和愤怒，其实也只是不够牛而已。

当然，我说的这种张扬也好，愤怒也罢，主要是反映在你的工作和技术领域，如果你把这些情绪完全带入到生活中，那就比较危险了，严重时会导致人见人踹，花见花败，除了至交好友，少有人愿意搭理你，

如果再不悔改，可能会像你预想的那样，再也没有人愿意和你交流。这种结果很容易理解，换位思考一下就好了，如果你说的每一句话和每一个观点都遭到聆听者的反驳和质疑，那这种交流就没法进行下去，如果是工作中的讨论还可以商榷，真理越辩越明嘛，但是在生活中，很多时候大家就是聊聊天，舒缓一下心情，比如对方见面问，“你好！”你说，“你怎么知道我好？”这时候板砖就飞过来了。

程序员有性格是好事，但是谁也不可能永远都对，多聆听少说话不是坏事。我有一个朋友，我知道他学识非常渊博，但是每次大家聊天或讨论问题，他都处于一种聆听的状态，频频点头若有所思，但是如果你征求他的意见，就会发现他总能一针见血一剑封喉找到问题的关键。我就知道，他一直在从别人的谈话中获取自己需要的知识。当然，他不是程序员。

程序员一般都比较自负，我年轻也是一个德行，谁要是对我的代码说三道四，恨不得掏刀子和丫拼了，但是慢慢你就会意识到，不停的反驳别人不会证明自己的聪明和独立思考，正确的讨论技巧和解决问题才是王道，比如有人提出了一个创意，你觉得有问题，可以这样说，“我说Mac君啊，你今天的文章总体来说还是不错的，照顾了大多数的读者，观点也比较新颖，但是呢，似乎没有重点还多了点狗血内容，如果能够……一下，就更完美了。”这种说法基本上会让你避开板砖与西红柿齐飞的场面，并造成大家在同一战壕的“假象”，有利于迅速有效的解决问题。

当然有人会说，乔布斯骂人都是直接说人家狗屎的，哪那么多弯弯绕呢？其实很简单，因为他是乔布斯，你不是。

总之，程序员要保持自己的性格、激情、愤怒，这样你才能写出传世的代码，同时也要温和、有理有据有节的与别人探讨问题，还要有健康的生活和好的娱乐活动。除了技术书籍，多看一些人文类的著作，有助于完善自己，善待他人！

今天似乎写了一堆碎碎念，也不知道是否回答了这位读者的问题，就这样吧，你们可以认为我说的都是错的！

程序员如何提高英语阅读水平

【发布日期　2013年4月16日】

问题：作为一名程序员，虽说每天都在和英语打交道，但是当看到一篇英语文档或者英语技术文章的时候还是比较头疼，理解他们的意思也只能是20％。尤其是使用google搜索的时候，很多问题解决办法都是英文的，还有一些国外比较有名的网站比如stack overflow，上面也有很多学习的资源。怎样才能让自己顺利阅读这些技术文章呢？

回答：其实学英语和其他技能没什么太大区别，无论是你想在英语阅读、口语或写作方面提升自己，都需要进行长期的不间断的练习，坚持一段时间后（时间长短根据你自己的效率、每天用时、频率都有关系），你会发现自己的水平自然就提升了。举个例子，以前写博客似乎是最难坚持的，但是如果你每个月都能写一篇略有价值文章的话，5年就会有60篇高质量的博客，你几乎都能集结出书了。到了微信时代，似乎写微信公众平台是最难坚持的，但是如果你能坚持一年，那就有300多篇文章，于人于己，都是一份宝贵的财富。

作为程序员，英语阅读能力是最基本的要求，相对口语和写作来说也是最容易达到的，因为计算机类图书的那些常用单词就那么多，多读几本英语类技术图书，想不认识都难。

一个相对容易坚持的办法就是，找一本和当前工作相关的、急需的技术图书，每天拿出一小时阅读，不认识的单词，如果不影响阅读可以不查词典。如果某个单词多次出现，那么就该查下词典并计入生词本。如果你能坚持查阅英英词典，那么提高就会更快了。

每天坚持一小时，这本书读完，你就会发现自己的阅读能力提高了一大截。

这样看来提高阅读能力似乎很简单，但是，问题的关键是，你能否坚持下来，尤其是阅读初期，你只能看懂20％，需要不停的查字典，单词记了又忘，几个星期过去了，进展依然缓慢。这时候最容易放弃，你会痛苦的怀疑自己，我究竟是不是这块料。

毕竟人类的大脑都是倾向于舒适和懒惰的，谁都知道看美剧刷微博，是容易的愉悦的。学英语学编程是痛苦的，有时候你甚至会有意识的去避免开始这件事情，先干点这个，再干点那个，熬到最后，发现没时间了，再拿起书来读一会，困了，今天先睡吧。就这样，一天天很快

过去了，你发现自己似乎每天都在坚持，但依然没效果。但事实是，你既没坚持，也没效率，这根本不是刻意练习。

提高英语阅读能力这事，比提高英语口语和写作能力容易多了。如果程序员缺乏英语阅读能力，这将是你非常大的一块短板，如果意识到了，尽早补上。

另外说一点，英语翻译对提升英语阅读有一定的帮助，但这个要求比较高，必须要查字典，遣词造句，力求原汁原味的表达作者的意图。翻译并不是提升阅读能力的捷径。我曾经翻译过很多文章和半本书，没觉得自己英语水平有很大提升。

最简单的坚持，最后的结果都是惊人的。这世界上一直存在一条路，让我们的能力从平庸到杰出，这条路漫长而且艰辛，只有少数人愿意走下去，所以，优秀的人永远是少数。

普通人之殇

【发布日期　2013年3月25日】

什么是普通用户，我对普通用户的定义是，我们不会成为乔布斯或比尔盖茨，我们坐飞机不会掉下来，我们中不了几百万的彩票，我们当中只有很少人能够创建自己的公司，我们取得的每一点进步都来自持续艰苦的努力……



但是，这并不妨碍我们成为人群中稍稍优秀一点的群体，比如我们工作稍微优秀一点，唱歌稍微嘹亮一点，踢球踢得更猛烈一点，用电脑用的更高级一点……我想表达什么意思呢？无论大家工作还是生活，总要涉及各种领域，如果每个领域都浅尝辄止，那你就真的成了普通用户了，普通用户是没法进入高级用户的圈子的，因为那些人讨论的东西你

不懂嘛，你也不能给别人提供更有价值的东西，那才是真正的普通用户之殇。

大部分人都是普通人，大部分人也成不了盖茨乔布斯，但我们可以成为高级用户。

在写这篇文章之前，我又为Alfred写了一个插件，叫做FindYYeTs，FindYYeTs是为Alfred开发的一款workflow，主要功能是检索YYeTs（人人影视）上最新发布的影视作品。

用法：通过option+space呼出Alfred，输入yyets all，查看YYeTs网站最近发布的影视剧；输入yyets科幻，可以检索标题匹配“科幻”的影视剧，同样是网站最新发布的，不是所有历史数据。用上下方向键或command+数字选中需要的文件，回车可以直接在默认浏览器打开。

如果你是个美剧爱好者，你应该需要这个workflow。

点击原文可以到Github上阅读这个插件的简要说明和源代码，下载workflow。有时候我想，在这样一个海量信息时代，是不是每个人都该学学编程呢？这样你自己就可以做很多事，那里是一个新的世界，钥匙就在你的手中。

去创业还是继续编程

【发布日期　2013年3月31日】

有人说现在是个创业的最好时代，有人说是最坏的时代，无论如何，都无法阻止有志中青少年投入创业洪流，这些创业者中有大量的技术人员，或者说有大量的程序员创业。这些在0和1的世界里游刃有余的技术高手怀着改变世界和发财致富的梦想，开始创业后才发现，现实世界并没有程序那么美好。旁观别人的创业和成功，总是充满传奇和荣耀，其实那些阴暗、琐碎和繁杂都隐藏在美好的东西后面。

在编程的时候，计算机是如此的忠实和温顺，你输入你的算法、判断和流程，编写计算机能够识别的程序，它就会严格执行这些逻辑，对的绝不会错，错的它会告诉你错在哪了。计算机几乎不会闹情绪，也没有低潮期，偶尔死机一次，重启一下又运转如飞。而在现实中创业呢？需要注册公司、租房、买设备、招人、裁人、谈判、决策、市场推广、产品销售，工商检不爽了要动之以情，员工闹情绪要晓之以理，一个决策不对不是客户跑了就是单子丢了，而且一旦开始创业，你发现你平时用来学习和提升技术的时间会被各种琐事毫不客气的瓜分，你担心自己最骄傲的技术也开始技不如人了。

怎么办？去创业还是继续编程？

我曾经见过很多优秀的程序员最终变成了一个出色的企业家。比如带我入行的师兄三无程序员（不是无证程序员，而是技术极其强悍，我们称之为无论操作系统、无论编程语言、无论数据库），后来创建了多家企业，而且技术也没扔，依然在编程。另一位我很熟悉的创业者，从程序员做起，最终掌管一家大型公司，成绩斐然。与上一个例子不同的是，后者完全放弃了技术。

国外的例子就更多了，比如苹果的沃兹、微软的盖茨和艾伦、Facebook的扎克伯格和Google的佩奇和布林，这些人无不具备超强的技术嗅觉和编程能力，在计算机发展的重要年代单枪匹马撼动世界，他们的起点都是程序员。

所以我的看法是，技术人员创业和继续编程并不矛盾，但凡在编程之余琢磨创业的，基本上都不是传统意义上的技术人员，他们在编程之余还想做一些更有挑战的事情，那就去做好了。结果无非有三种，真正对技术有追求的，会创业并继续编程；而更擅长设计、管理和经营的，

慢慢会放弃编程并转向一个更适合的领域；还有一部分发现完全不适合创业的，那就专心做技术好了，别担心浪费的那些时间，那些创业实践可能会给你带来更多的编程灵感。

而且，我觉得创业并不一定要完全从头做起，只要是在合适的环境里去做自己能够全盘掌控的事情，就算创业了。

去创业还是继续编程？如果你在问这个问题，我觉得除了技术之外，你该做点别的了。

趣谈个人建站

这篇文章是分四次完成的。虽然是技术文章，但我尽量把这件事写的轻松一点，大家读起来也更有趣。最后形成了一篇完整的趣谈个人建站。



2000年前后是第一波互联网浪潮，无论是幸与不幸，我的早期职业生涯都是从这波浪潮开始的，那时候很多ASP（Application Service Provider）厂商会给个人用户免费提供一些静态建站功能，大家可以写一些HTML+CSS+JS的页面传上去，算是早期的个人的站点，我记得自己的第一个站点叫做“雪域苍穹”，貌似取自一首流行歌曲的名字。无论是名称还是页面，现在看来都土的能掉出翔来，但当时的感觉是，这特么的太酷了。

后面做过一些个人网站，由于各种原因都关掉了。再后来开始写博客，很多人开始建自己的博客站点。站点不少，一直保持更新的倒没几个。我的想法是，专业的事就让专业的厂商去做吧，所以一直也没建个人博客网站，断断续续的在博客园和图灵社区写一些东西，也算是保持更新了。

终于有一天，微信公众平台来了，一个偶然的机会注册了MacTalk（原Mac技巧），之后一口气写了一百三十多篇文章，文字总数超过了我前几年的博客总和，而且保持了一定的文字水准（自以为-_-

#）。然后就有很多读者一直提醒我，MacTalk里的内容有一部分是技术性质的，有存留价值，如果能够进行查询检索，对Mac的新老用户都有帮助。我想了想也是，扯淡的东西估计没人愿意重复阅读的，技术类又很难记在当下，所以就准备开始着手建站，然后macshuo.com就建成了。下面我把整个过程写一下，供大家参考，另外，我只说自己的选择，不会去比对各种指标，比如Linode和国内VPS的优劣，Apache和Nginx的性能差异等等，如果你想了解这些东西，那就用Google百度一下。

搭建个人站点，大致需要做这么几件事情：

* 一台具备公网IP的服务器
* 安装操作系统，搭建环境
* 购买域名，域名绑定IP
* 部署应用程序

基本上这四套组合拳打完，你的个人网站就算建起来了，后续的事情就是添砖加瓦和蓬荜生辉了。

好吧我们依次介绍：

服务器

大部分公司都会有自己的服务器和公网IP，要么托管要么自建机房。但对于个人用户来说，就没必要费时费力做这个事情了，购买一个VPS（Virtual Private Server）即可。什么是VPS，建议大家去维基百科上查一下，简单来说就是你会拥有一台虚拟主机，除了看不见机箱之外，你可以像操作一台实体服务器那样操作它，独立操作系统和硬盘空间、独立内存和CPU资源、独立的执行程序和系统配置等，可以自己安装操作系统和软件，独立重启等等。

在VPS的选择上，我用的是Linode。Linode是一家来自于米帝的专注于提供Linux VPS的服务提供商，虚拟化技术采用了Xen，Linode的含义是Linux Node。注意，这里的操作系统是Linux，我推荐所有个人建站都采用Linux，不解释，如果你想采用Windows Server，后面的内容就不用看了。

Linode在国内外口碑都不错，价格适中，质量可靠，童叟无欺。Linode提供了各种Linux操作系统供选择，比如Ubuntu、Redhat、Debian、CentOS等等，装系统和重装系统都非常简单。

好，我们下面简单说一下步骤，访问：https://manager.linode.com/session/signup

填写邮箱、用户名密码，就算注册成功了，Linode会给你发封邮件确认，打开那个确认链接，大家就会看到下面这张图的内容：



Linode通过它的ticket system（一套支持系统）提供7x 24x 365的支持服务，看清楚，不是7x 24x 365的不停机服务，我现在特别烦一些企业客户，一谈就说永不宕机，特么除了上帝谁能保证永不宕机？时间长了自个都得宕！另外Linode还提供了4小时的免费试用服务，比较厚道，如果你试试觉得不爽还可以选择不玩。

选择继续，就可以选机房了，Linode目前提供了东京和欧美等地的机房选择，我选了东京机房，据说是针对亚太地区用户的需求新开辟的，速度很快。然后选操作系统，设置硬盘大小、root密码等，点击“Rebuild”，你就进入了VPS的控制台，等Host Job Queue的所有任务都是绿色的Success，就可以点击“Boot”，启动系统。然后找到Remote Access这个标签，点进去就可以找到这台服务器的访问IP，打开终端，输入ssh root@x.x.x.x，就可以登录系统了，看到了吧，very simple！

试用之后，如果你觉得可以，点击Account标签，完善自己的信息，选择服务器配置，支付信息，然后就可以完整支付流程了。

我选的是Linode 1024套餐（24GB DISK, 2000GB），按照年付费的话230刀左右，大家这两天赞助的碎银子，差不过够一年年费了：）支付方式包括Visa，MasterCard，American Express，只有要信用卡还是很方便的。

另外需要注意的一点是，拿到了IP之后，一定要在不翻##墙的情况下测试一下是否可以正常访问。我就遇到这个问题了，在国内没法访问，但是挂了VPN的就可以，我估计是哪个倒霉孩子以前用过，被墙之后不用了。

不得已我发起了一个Ticket（支持问题），说我在中国大陆不能访问这个IP，但通过VPN可以，那哥们响应倒是挺快，但显然不懂我朝行情，让我执行mtr——r x.x.x.x，mtr可以结合ping、nslookup、tracert诊断

网络传输问题。我只得把数据返给他，结果人家还要其他数据，我就不耐烦了，用蹩脚的英文给丫解释了一下什么是伟大的墙，基本意思就是少特么废话，赶紧给我换个IP。那哥们看我气势挺盛，赶紧给我换了个IP，我一试没问题了，说了声三克油，他说威尔卡姆，这事算结了。两人共交手五个回合，用时2小时，效率还可以。

好，服务器部分就介绍到这里。以下是我的Linode推荐码，如果大家要购买Linode服务，可以用这个链接。http://www.linode.com/?r=6bd100da844d8d2c191680a4792610467ce9052

搭建环境

我选用的服务器是Ubuntu12.04，64位。以下内容均基于该环境描述。

拿到了主机IP，你就算拿到了新房的钥匙，但是离入住还远着呢，因为你那个主机现在就是个毛坯房，除了进去看看，什么都不能干。好，下面我们做一下简装修。

1、创建用户

第一次登录需要root用户，什么是root？root就是整个Linux操作系统最牛逼的主，他想干嘛就干嘛，他想删谁就删谁，他是光他是电他是唯一的神话，他就是我朝就是我D，所以非常危险，你们懂的。如果用root执行一下rm -rf，那整个锡安就会被抹掉，尼奥也拯救不了，如果root愿意，他可以抹掉你曾经存在过的所有痕迹。所以，我们不能没事就用root进去耍，为了解决这个问题，我们必须要建立一个agent，平时是普通用户，关键时刻充当root的角色。

具体操作如下：

首先用root登录系统

ssh root@x.x.x.x

创建一个新用户，用户名随你喜欢，比如叫做mactalk

adduser mactalk

按照提示信息输入密码和相关信息，就可以完成操作。完成之后系统就会自动建立/home/mactalk路径。

然后是授权，输入

visudo

在编辑器中找到如下内容：

root   ALL=(ALL:ALL)ALL

在下面加一行

mactalk   ALL=(ALL:ALL)ALL

通过ctrl+x保存退出即可。然后就可以退出root，用mactalk重新登录（ssh mactalk@x.x.x.x），登录进来默认目录在/home/mactalk下，当你想行使root权限时，请在命令之前增加sudo，按照系统提示输入密码即可执行操作。

2、选择**shell**

用户建好了，下面我们为用户选择一种shell，估计小白看到这个又毛了，啥是shell？

shell就是Linux的一个外壳，你理解成衣服也行。它负责外界与Linux内核的交互，接收用户或其他应用程序的命令，然后把这些命令转化成内核能理解的语言，传给内核，内核是真正干活的，干完之后再把结果返回用户或应用程序。比如你对shell说，“你好”，shell就跑到内核那说，“老大，有人问候你呢”，内核就不耐烦的说，“有事说事，我特么忙着呢”，shell就把这条信息反馈给你，大致就是这样。以前讲Mac技巧的时候，经常跟大家说在终端里输入一些命令，那就是Mac的shell，都是一脉相承的。

Linux提供了很多种shell，你要问我为什么要有这么多，我只能告诉你，你为毛同类型的衣服有那么多件？花色，质地还不一样。写程序比买衣服复杂多了，而且程序员是不惮于把事情搞复杂的，牛程序员看到不爽的shell，就会自己重新写一套，慢慢形成了一些标准，常用的shell有这么几种，sh、bash、csh、zsh等，想知道你的系统有几种shell，可以通过以下命令查看：

cat /etc/shells

这些shell我就不解释了，维基百科和百度百科都写的很清楚，总之，坊间流传，普通程序员用bash，文艺程序员用zsh，XX程序员直接用原生的sh，我建议大家文艺一点，用zsh好一些，功能也最强大。目前各个版本的Linux默认的shell都是bash，如果你想用zsh，需要安装一下，如下：

sudo apt-get install zsh

具体的配置我就不介绍了，感兴趣的读者，可以参考http://leeiio.me/bash-to-zsh-for-mac/

3、通用工具

介绍几个简单的工具，建站必备。

* wget，命令行下载工具，安装sudo apt-get install wget，使用方式后面会介绍。

* tmux，一个优秀的终端复用软件，类似GNU Screen，但来自于OpenBSD，采用BSD授权。使用它最直观的好处就是，通过一个终端登录远程主机并运行tmux后，在其中可以开启多个控制台而无需再“浪费”多余的终端来连接这台远程主机。好吧，这句话有点绕，简单说就是用tmux打开的会话可以一直驻留在服务器上，下次去看时还是上次来的样子。就像你是某个酒店的VIP客户，住完之后不会人走茶凉，也不会断电，下次去时茶还热着，灯也亮着，就这样。

安装方式sudo apt-get install tmux，对使用方式感兴趣的读者去查一下吧，中文介绍很多，记住，热键是ctrl+b。

* vim，在Linux上少不了编辑文件，我推荐Vim和Emacs，一个是编辑器之神，一个是神的编辑器（或者是伪装成操作系统的编辑器），我是Vim党，目前在学习Emacs。我之前写过一个Vim系列，有兴趣的可以去看：http://www.cnblogs.com/chijianqiang/tag/vim/

从原理到配置、使用都有非常详细的介绍，那也是个大坑，还没写完，但写了MacTalk就变成顾此坑失彼坑了。

差不多就这几个，其他的工具随用随装吧。

域名和DNS

服务器和环境构建都写完了，今天介绍一下域名和DNS的那点事儿，稍微复杂一点的软件部分放在最后说。

域名是什么东西呢？就是一个网站的标识和入口，由“.”分隔开的字符串构成，洋名叫Domain Name，比如苹果公司网站的域名就是apple.com，在浏览器地址栏输入这个域名，就可以访问苹果的网站了。为什么要有域名呢，有了公网IP，不就可以访问网站了么？

咳，这么说吧，如果有人说找个地方聚聚，你说，咱们经度116.46、纬度39.92，不见不散！地方倒是对，但是估计实名菜刀和无名臭鞋就飞过来了。IP地址就是你的服务器在互联网世界的经纬度，域名就是对应IP的门牌号码，就像人们能记住门牌号记不住经纬度一样，在网络世界里，大家都是记域名的。与现实世界不同的是，门牌号和实际地址正常情况都是一对一的，而域名和IP地址是多对一的，也就是说，只要你有一个公网IP，就可以申请多个域名，对应多个应用，非常方便。

现在你知道了吧，要建站，必须要有域名。能够提供域名的厂商很

多，国内外都有。不过我强烈推荐大家购买国外厂商的域名，免去提交材料和备案之苦，国外动动鼠标和小手分分钟搞定的事情，国内要提交各种材料、备案、定期监管balabala……具体差异大家看看www.apple.com.cn和www.apple.com两个网站的底部知道了，苹果中国的底部有“京公安网安备11010500896|京ICP备10214630”，再看看米帝的网站，毛都没有，都是自己的网站信息，我们只能说，米帝的监管制度太不健全了，真为他们捉鸡！

国外的域名厂商推荐www.godaddy.com和www.name.com，都不错。我使用的是Godaddy。

Godaddy是全球最大的域名注册服务商，全球市场占有率超过30％，一般情况下不会被我朝屏蔽。如果屏蔽了Godaddy，会导致在大陆无法访问全球近三分之一的网站，所以相对安全，相对，你懂的。另外Godaddy开始支持支付宝了，对我朝臣民来说付费变得方便无比。

在域名选择上，最好满足这几点要求：有意义、好记、简短，另外尽可能使用com（通用顶级域名）。申请步骤也很简单，访问www.godaddy.com，在搜索框输入你想要购买的域名，点击搜索，你会看到这个域名的具体信息，是否被使用，相关域名，价格等信息，域名后缀一般有com、net、me、us、info等，建议选com，不建议选info，据说info结尾的网站大部分是垃圾网站，会被搜索引擎屏蔽。

选好域名后，点Add，加入购物车，如果不需要其他服务，一路Continue即可，最后设置支付信息，支付宝在最后一列，然后“Place Your Order”，根据提示注册和支付即可。价格与域名信息相关，一般几美元到十几美元不等。

购买完成之后，进入Godaddy的域名管理控制台，找到你购买的域名，把域名和你的IP地址绑定起来，就可以通过域名访问你的网站了。这里就涉及到DNS了。

DNS的洋名一般说成Domain Name System，就是给域名提供服务的。光有域名没用，还得有相关的服务能够把域名解析成IP地址才行，DNS就干这事。DNS的扩展性非常好，不依靠单一的巨型主机索引，而是通过分布式系统提供服务，全球能够提供DNS服务的服务器多如牛毛数不胜数，但大哥级别的根服务只有13台，目前的分布是：主根服务器美国1个，设置在弗吉尼亚州的杜勒斯；辅根服务器美国9个，瑞典、荷兰、日本各1个，旗下是各个级别的域名服务器。

DNS的工作方式挺复杂，有兴趣的直接去Google吧，我简单描述一下，大致的场景是这样的，比如小明是个DNS服务器，有一天你想访问macshuo.com，就问，“小明啊，我想去听听MacTalk，怎么走呢？”，作

为忠诚的DNS战士，小明的服务态度是值得赞赏的，如果他知道的话，就会立刻告诉你地址，比如从这个街区左转左转左转再左转balabala……如果丫不知道也没关系，他会反馈给上级主管，说“最近有人新建了个MacTalk网站，我这还没记录地址，你晓得不？”，上级部门如果知道就会把地址发给小明，说“你丫长点记性，把这地址记下来行不？别特么老问了，最近自媒体网站可真多啊，哎……”，这样小明就会把地址告诉你，并且用心的把这个地址记到自己的小本本上，下次有人再问，就直接告诉人家答案了。如果上级部门也不知道，那就继续问，直到反馈到根服务器为止，反正只要你注册了，总能找到。差不多就这样吧。

Godaddy默认提供了DNS服务，点击DNS Manager，在弹出的设置面板中设置你的网站IP即可，具体的图文教程网络上很多，我就不费那事了。但是我的使用结果是，Godaddy提供的DNS在国内访问不太稳定，时不时就不能访问了，具体为啥我也不清楚。因为购买了Linode服务，我最后采用了Linode的DNS，具体的做法是：

1. 登录linode.com，点击DNS Manager标签，进入管理控制台
2. 选择Add a domain zone
3. 填写域名、邮件地址和IP
4. 点击Add a Master Zone，就算完成了

最后一步就是在Godaddy的控制台里设置一下Linode的DNS服务器，很简单就不描述了。

应用程序和部署

经过前面三篇系列文章的介绍之后，我们现在房子也有了，做了简装修，还申请了地址和门牌号，最后一步就是入住，也就是部署你的应用程序。

针对网站提供的服务不同，需要不同的技术选型，我的需求就是做个风格简约的博客，用来存放MacTalk的文章，同时有个地方能够随意发表一些个人观点，就这么简单，所以针对这个需求进行选择即可。大家将来建站的时候也是一样，明确自己的需求，不要为未来买单，尽量搞的轻量级一些，最忌讳给的是龙套的钱，您自个却按照男猪脚进行角色扮演，不提倡。

搭建轻量级的博客不建议使用.Net或JavaEE的技术，这些技术都比

较重，必要性不大。Php、Python、Ruby相关的框架都是可选的技术。因为我对Python相对熟悉一些，最初想找个开源的Python Blog框架，不过后来综合对比了一下，发现在个人博客领域，WordPress基本上无出其右，技术成熟、安装方便、性能稳定、插件众多，实在是居家建站、个人扯淡之必备良药，就是它了。

在确定了基本需求和工具之后，我们看看涉及到哪些技术：

1、Nginx Nginx是一款高性能的HTTP服务器软件，由俄罗斯的一位大牛Igor Sysoev开发的，源代码以类BSD许可证的形式发布。Nginx的设计非常轻量级，由内核和模块组成，内核微小简洁，模块功能强大，静态编译。Nginx做的事情简单来说就是，接收客户端（浏览器）的HTTP请求，然后通过映射机制把不同类型的请求交给不同的模块去处理，比如html、图片、css等可以交给静态资源模块处理，还可以做压缩、缓存等，php、python等类型的请求则交给FastCGI模块去处理，完成业务逻辑。

什么是FastCGI呢？这玩意就等于是HTTP服务器和动态脚本语言通信的接口，就像一个粘合剂一样把HTTP请求和动态脚本处理整合在一起，顾名思义，处理速度非常Fast！

Nginx可以说是HTTP服务器软件市场的新贵，目前国内很多大型网站都采用了Nginx作为默认的web服务器，比如阿里、腾讯、新浪等等，国外就更多了。当然，在Nginx未涉足江湖之前，这个领域的大哥叫做Apache，那时候几乎所有的HTTP Server都是清一色的Apache，一时之间风头无两。但是这哥们大哥做久了就不思进取，跟国内很多大佬一个德性，没竞争对手时就特么不知道改进，直到Nginx出来抢了丫半壁江山，现在知道努力了，当初干嘛去了？

那么Nginx比Apache优秀在哪呢？

* Nginx的所有模块都是全静态编译的，启动Nginx后，Nginx的模块被自动加载，静态库执行效率更高。

* Nginx支持epoll（Linux系列）和kqueue（BSD系列）I/O事件通知机制。完，又特么出现两个名词！这让人情何以堪、文何以完啊？简单说说epoll吧。epoll是Linux2.6正式引入的提高网络I/O的处理方法，它的几个优点是：单一进程打开的FD（文件描述符）数量仅受限于操作系统，1GB内存的机器上大约是10万左右，这一点大大提升了处理海量请求的能力；采用共享内存的模式避免内存拷贝；随着打开FD的数量增加，I/O效率不会线性下降。总之，大家知道epoll很牛逼就是了。

* Nginx支持多进程的工作方式，Nginx启动后会有一个master进程，多个worker进程。worker进程一般对应服务器的CPU数量，你有个8

核的CPU，最好把worker设置为8。master负责接收外界信号，并向worker发送信号，监控worker的运行状况，当worker挂掉的时候，启动新的worker。写到这我发现，这特么活脱脱就是一个地主老财打压长工的模式啊！

尤其是Nginx的不中断重启机制，当系统配置变化需要重新启动Nginx时，我们就给地主（master）发个消息，说这批长工（worker）太老了，都得换掉，你看着办。地主收到消息后就开始偷偷雇佣新的长工（worker），然后假惺惺的告诉老长工，把手头的活干完就行了，别太累了，啊。这时候如果有新的请求，就会交给新长工干，等所有的老长工把活都干完了，就直接fire，绝不留情。这样，整个服务无中断重启过程就完成了，就特么一个字，黑！

当然长工（worker）也不是省油的等，他们干活的过程非常复杂，会用到我们上文中提到的epoll机制，如果有人感兴趣，以后再讲吧，这么写下去这个系列就没完了。

Nginx功能非常强大，一本书也写不完，我简单就说这么几句。老话，有兴趣的，用Google百度一下！

大概了解了Nginx的工作机制，下面安装就比较简单了，Nginx可以编译安装，也可以在线安装，对于普通用户来说，使用apt-get在线安装即可，省得自己去找依赖关系。

```
#安装
sudo apt-get install nginx
#启动
sudo service nginx start
```

如果安装和启动都没有问题，我们再调整几个参数就可以了。

找到/etc/nginx/nginx.conf，做以下几个改动：

* 把worker_processes设置为服务器的CPU核数
* 在event里增加use epoll
* 把worker_connections的值设置大一点，如果是1G内存，不要大于100000/worker_processes。

其他的采用默认值即可，然后重新加载参数：

```
sudo nginx -s reload
```

好，Nginx就算妥了，后续在安装PHP和WordPress时还要做一些配置。

我有时候觉得，当我们在计算机领域遇到问题的时候，总会出现一些技术神山上的神人，他们时不时会俯视一下凡人的IT世界，高兴了就顺手解决几个bug，发明几个新玩意，epoll和Nginx就是这些玩意，我们用好就已经心满意足袅。

2、MySQL

MySQL是应用最为广泛的开源数据库，这个没什么可说的，非常成熟的技术，直接安装即可：

```
sudo apt-get install mysql-server
```

安装过程中，MySQL会提示你设置root密码（root的作用参考之前介绍的，把操作系统换成数据库即可）。如果安装时没设置密码，等MySQL起来后用mysqladmin改也行，用sql改也行，简单不啰嗦。

3、PHP

WordPress是基于PHP开发的，所以我们得为WordPress准备好环境，安装PHP。

```
sudo apt-get install php5
sudo apt-get install php5-fpm
```

php5-fpm是PHP FastCGI的实现之一，能够更好的管理PHP进程，控制内存使用，平滑重载等，现在我们都用它！

下面做一点简单配置，打开php.ini文件：

```
sudo vim /etc/php5/fpm/php.ini
```

找到cgi.fix_pathinfo=1这一行，把1改为0。值为1时，php的解释器会尽可能的去解析客户端请求的文件各种类型，这会引发一些安全漏洞，设置为0时，解释器只会去解析特定的文件类型，设置为0是一种相对安全的处理策略。

修改www.conf：

```
sudo vim /etc/php5/fpm/pool.d/www.conf
```

把listen =127.0.0.1:9000修改为listen =/var/run/php5-fpm.sock，前者是走TCP socket，后者是Unix domain socket，如果服务都在同一台机器上，建议使用后者，效率更好一些。

重新启动PHP，这部分的配置就算完成了：

```
sudo service php5-fpm restart
```

4、WordPress

好的，我们从后场断球后左冲右突，盘过对方和我方的所有进攻和防守队员，来到球门前面，发现就差最后一关：WordPress。下面我们看看如何安装和配置WordPress。

首先下载WordPress的最新版本，我用了中文版，下载和解压缩：

```
wget http://cn.wordpress.org/wordpress-3.5.1-zh_CN.tar.gz
tar -xzvf wordpress-3.5.1-zh_CN.tar.gz
```

在MySQL中为WordPress创建用户和数据库，这部分很简单就不描述了。我们设定数据库为wordpress，用户名为mactalk，并且把数据库编码改为UTF-8。

在解压好的wordpress文件夹下，执行：

```
cp wp-config-sample.php wp-config.php
vim ~/wordpress/wp-config.php
```

按照文件内容注释填写数据库名称、用户名、密码、数据库编码使用UTF-8，然后保存退出。

为wordpress创建www文件夹，并且把完整的wordpress目录复制到www文件夹下，并设置相关权限：

```
sudo mkdir -p /var/www
sudo cp -r ~/wordpress/*/var/www
cd /var/www/
sudo chown www-data:www-data *-R
sudo usermod -a -G www-data username
```

安装php的MySQL驱动

```
#sudo apt-get install php5-mysql
```

设置虚拟主机：
在/etc/nginx/sites-available下创建文件wordpress

```
sudo vim wordpress
```

内容如下：

```
server {
    listen    80;

    root /var/www;
    index index.php index.html index.htm;

    #根据IP或域名自定义
    server_name 3.3.3.3;

    location /{
        try_files $uri $uri//index.php?q=$uri&$args;
    }

    error_page 404/404.html;

    error_page 500502503504/50x.html;

    location =/50x.html {
            root /usr/share/nginx/www;
    }

    # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9$
    location ~\.php${
        #fastcgi_pass 127.0.0.1:9000;
        #With php5-fpm:
        fastcgi_pass unix:/var/run/php5-fpm.sock;
        fastcgi_index index.php;
        include fastcgi_params;
    }
}
```

这个文件的作用就是把Nginx和WordPress粘合在一起，接收客户端的请求并反馈响应结果。有几点要注意的是，root设置为/var/www/，index部分增加index.php，fastcgi_pass对应之前设置的unix socket：unix:/var/run/php5-fpm.sock。

为wordpress文件建立软连接：

```
sudo ln -s /etc/nginx/sites-available/wordpress /etc/nginx/sites-enabled/wordpress
```

最后重新启动nginx和php5-fpm，就算大功告成了：

```
sudo service nginx restartsudo
service php5-fpm restart
```

如果一切正常的话，访问你的域名或者公网IP，就可以看到wordpress的提示页面，根据信息提示初始化数据库，创建管理员，基本框架就算建好了，之后就是完善和优化，比如性能优化、主题选择、配置信息、插件选择、扩展开发等等，大家慢慢体会吧。

这个系列就算完结了，我发现写这种文章想做到好玩好看、还能言之有物把事说清楚，挺难也挺累，好在结了。

第一个十年我才华横溢，“贼光闪现”，令周边黯然失色；第二个十年，我终于“宝光现形”，不再去抢风头，反而与身边的美丽相得益彰；进入第三个十年，繁华落尽见真醇，我进入了“醇光初现”的阶段，真正体味到了境界之美。

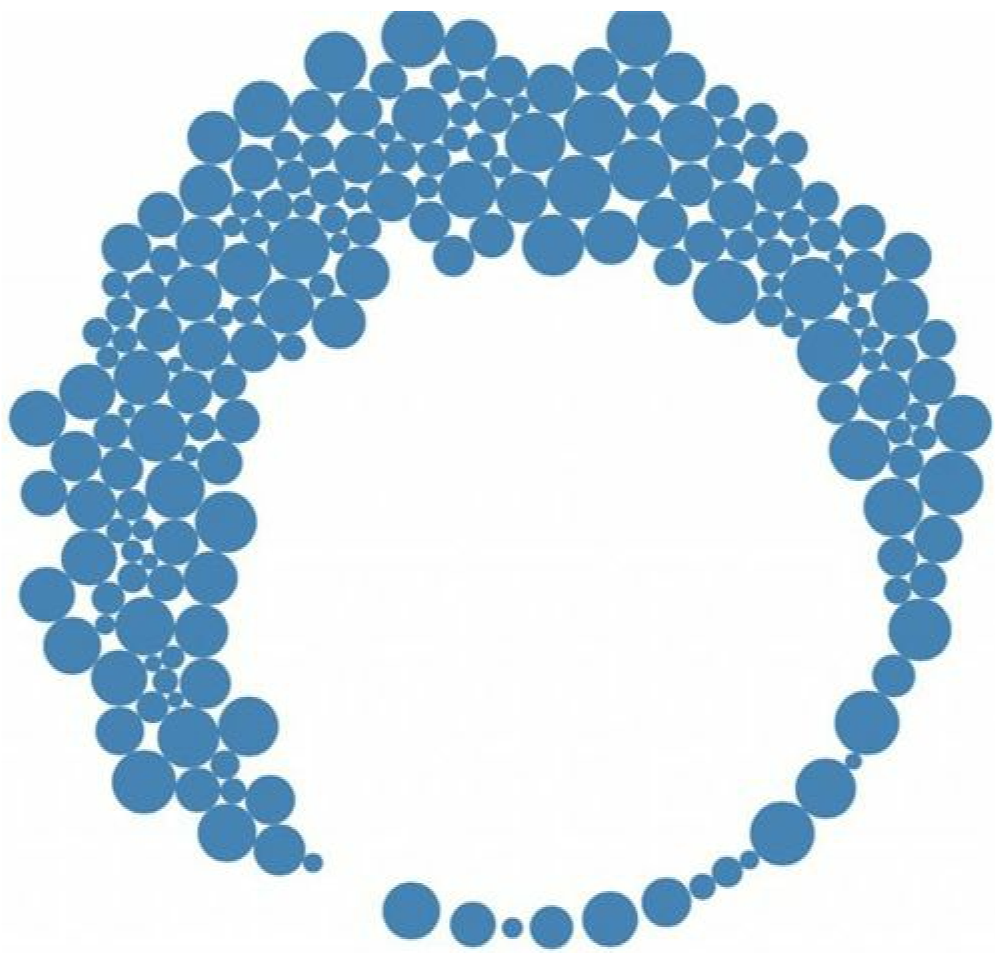
——台湾作家林清玄

人生元编程

【发布日期　2013年6月24日】

如果你是个诗人，那你的人生就是一场风花雪月的事；如果你是个演员，除了学好《演员的自我修养之外》，你的人生就是一场接一场的剧本；如果你是个侠客，对不起，现代没有侠客，你会被抓起来进行休假式治疗；如果你是个程序员，好吧，你的人生将由一行行飘逸的代码和捉摸不定的bug组成，所谓编程人生，就是你的一生已经与编程密不可分，为代码欢笑，为bug忧伤。

那一年你初入江湖，你不懂什么是汇编什么是语言，你搞不懂Lisp和Smalltalk的区别，为什么C++比C多了两个加号就成了对象，2000年以后那么多人都在用Java，现在却说Objective-C是最贵的语言，这特么又是为毛？



你对着大海说，我要学尽天下武功！大海对你说，你算术不太好。

孩纸，世界上的编程语言成百上千，常用的也有数十种，光学会这些语言你的时间就得用微积分计算，学完以后估计手抖的都敲不动键盘了，哪还有时间创造奇迹？更别说除了语言你还得掌握前端后端UI体验，这个数据库那个操作系统balabala……

你在知道了这些真相以后，依然痴心不改，抹干眼泪冲到编程兵器排行榜“TIOBE Index”面前，挑选了前十名开始勤学苦练。你在满天星斗的夜色中编写C程序，在清晨的微光中调试算法，上午你敲打键盘输出日志，中午吃完五又四分之一口米饭之后就匆匆离开，因为你要去看看系统为什么崩溃……你学会了五种语言、三种操作系统和四种数据库，你写了一个MIS两个OA三个App，你觉得你开始了编程人生，其实是你的人生被编程了，你被代码驱动和驱赶，你变得疲惫不堪。

这次你在清晨的寒风里对着高山说，这特么是为什么？高山对你说，因为你不懂元编程！

好吧，扯了这么多其实是想和大家谈谈元编程的事儿。元编程？估计小白一听又懵了，啥是元编程呢？与云计算、大数据不同，元编程并不是一个抽象的概念和名词，这里面代表了很多务实的技术，相伴而行的概念还有元数据。

元在英文里就是meta，元编程就是meta programming，元数据就是meta data。元编程就是能够操作代码的代码，元数据就是能够描述数据的数据。

听完这样一个介绍，大家是否更加晕菜了呢？如果回答是肯定的，那么效果达到了。

在接着介绍元编程之前，我们先看一下代码的世界。如果把代码比作一座小镇，那么其中的类、函数、方法、变量、代码块、宏，就是小镇上安居乐业的居民，他们相互协作，相互依赖，一起建设着有XX特色的美好家园。

在能够支持元编程的语言世界里，你可以和这些居民打招呼，还可以进行内省（introspection），获取其自身的一些信息和行为，甚至你能够为这些居民动态增加一些能力和行为，或者在这些居民奔跑的时候改变他们的行为，或者创建一些新的居民。这样的语言有Ruby、Python等。

在不支持元编程的语言世界里，大家分为两个状态，编译时和运行时，一旦编译器完成了自己的工作，这些方法和函数就看不见了，他们

成为内存中的幽灵，你只能通过固定的方式使用他们，而无法获取他们自身的信息。当然，即使是这样的语言，为了增加编程的灵活性，也通过各种方式来提升元编程的能力，比如Java和C#笨手笨脚的使用反射方式，C++则通过模板方式，但古老的C就无能为力了，因为他没有元编程能力。

现在我们就知道了，编程语言虽然各有侧重，但是语言和语言之间的能力和特点区别还是很大的，不管你现在使用的是什么语言，我都建议你们去学一门具备原生的元编程能力的语言，比如Ruby、Python、Lisp、Objective-C等。

我第一次接触元编程和元数据还是在一家外企，那家外企的名字和火箭有关，他们有很多年纪一大把的老程序员，据说是制定corba标准的牛人，他们在这个火箭公司开发了一套分布式的软件平台，名字不能提，因为老外的版权意识太强袅。我一位前同事移民国外，只是在自己的开源项目引用了一点平台文档，结果一纸法院传票追杀到异国他乡，而且直接导致这个同事的上司被辞退。“好吧，上司不是我，不过我当时确实想过，如果我引用了他们的代码，也许会见到真的杀手吧。”

这套平台的持久化、权限和业务逻辑引擎都采用了元编程和元数据的方式实现，实现语言是Python，当时看到那些优雅的代码，我再次感受到编程的魔力，原来代码还可以这样写！我在那个外企的两点收获，第一是平台和元编程，第二是版权意识。后来当我有机会主导从头构建一个软件开发平台的时候，我吸取了这些思想和经验，基于元编程的思路构建了平台组件数据字典，你可以编写少量代码或不编写代码就生成各种业务应用，这就是操作代码的代码，描述数据的数据。

这时候就有童靴问了，你啰哩啰嗦扯了这么多元编程，干嘛标题叫做人生元编程？

因为无论是编程还是人生，都特么是相通的，想清楚了这一点，你就会觉得百无聊赖，因为万事万物要么是熊样要么是鸟样，都脱不出那个框框。具备元编程的语言就具备更强大的操控自己的能力，可以自省，可以反射，可以动态改变和控制自己；具备人生元编程能力的人，同样有自省能力，随时检查和控制自身的情绪和行为，思考自己的想法，改变大脑的动机。

举个简单的例子，当你的理智告诉自己9点就必须开始看书学习的时候，你的大脑会对你说，“亲，可以再看会电视呦，你看沙发都这么舒服……”

缺乏元编程能力的回答是“那……就再看会”，具备元编程能力的回答是“滚！”

如何提问

【发布日期　2013年3月22日】

现在MacTalk的读者越来越多，提问的也越来越多，但是好的问题却凤毛麟角，有些问题你几乎不知道要问的是什么，所以也无从答起。

提问和回答是交流最重要的部分，一个好的问题能够让提问者和回答者都得有收获。我在2005年左右，与美国程序员共同维护一个平台级产品，邮件往来必不可少，当时我就发现他们提的问题或bug都非常规范，每个bug都有清晰的标题，正文是环境描述、已经采取了什么措施、结果、日志、Core dump、截图等等，读完邮件你就能很清楚对方想要表达的意图和希望你能提供的帮助，而且你也知道该做什么，如何回复等等。



很多人说中美技术人员在创意和创造方面相差甚远，其实差距是全方面的，不仅仅是技术，还有文化、氛围、教育等等，这个扯远了……

那么就技术问题而言，如何去问一个让双方都满意的好问题并最大

程度的得到回复呢？大好人生，谁也不愿意为一个烂问题浪费时间。

简单总结一下，如果你按照以下步骤进行，相信提出的问题会更靠谱一些，提出好的问题是提升的第一步，其实这个过程在提问之前已经开始了：

1．遇到问题不要急着问别人，在时间允许的情况下看是否自己能够解决，一方面锻炼自己分析问题和解决问题的能力，另一方面，一旦问题解决了，问题就不是问题，而是你的经验和知识库。况且现在互联网有那么多的技术资料和各类问答网站，想碰到一个别人没碰到的问题，已经非常困难了。

2．如果做了努力依然不能解决，或者客观条件不允许你自己解决了，那么首先要选择提问对象，不管是是现实中的大神，还是网络上的牛人，确保他是你所知道的最佳解决人选。

3．你需要一个好的标题，用清晰的短句描述你遇到的问题

4．至关重要的正文

（1）用清晰的语言描述你遇到的问题

（2）提供软件环境，包括操作系统、数据库等相关软件及其版本号

（3）问题是否可以重现，采用什么方式重现

（4）采用了什么措施解决问题，最终结果（可提供日志、程序、截图等描述）

（5）尽可能提供问题相关的可分析文件，包括日志、截图和Core dump等

（6）不要长篇大论，简明扼要，描述主要问题

最后，不要忘了说请和谢谢，毕竟你需要别人帮助你解决问题，没人欠你什么。

如何学习一门编程语言

【发布日期　2013年6月24日】

关于学习编程这个主题，有各种读者多次要求写一写，而且要求文艺的写、抒情的写、充满社会主义特色的写，要做到：问题看起来巨复杂，读起来巨简单，学起来巨容易！看把你们惯成什么样子了，Mac君你继续去面壁吧。

好吧不管他，我们接着聊。

如果你准备未来投身到IT江湖从事编码这份有前途的职业，学习一门语言显然是远远不够滴，就像你初入江湖，告诉别人，

“兄弟只会太祖长拳！”

问，“Level可及乔峰？”

答曰，“不及万一。”


Lua
Microsoft
ASP.net
CF
C#
php
BASH
Java
PROGRAMMING REPUBLIC OF
PERL
Self
ERLANG

人家一看你就是P2的命，PK时一个大招直接秒掉。写到这我想起了一个叫做冰河的兄弟，也是奇葩一朵，在程序语言方面一生只爱Lisp（Lisp号称编程语言的祖宗），在人类语言方面则除了中英文，还在同时学习意、法、西、德四门语言，而且不是随便学学，而且神志没有错乱，这一点让我简直佩服到逆天，一门英语已经从初中折磨我到现在了，在人类语言层面，我常常是被秒杀的。

所以，如果编程有可能成为你的职业，那么5-10年的学习和实践时间是需要的，因为你可能要学习编程语言、操作系统、算法、数据库（Sql的NoSql的）、Web开发等等，还有各种数不清的引擎和架构，特别令人发指的是当你熟练的掌握了一门技术之后，就会有位赤脚大仙走过来告诉你，孩纸，你学的技术已经不是方向袅，然后在你绝望的眼神里飘然离去……写Java的兄弟感受一下……

如果你的职业发展与编程无关，只是想学习一门语言磨练人生意志，那么这事就比较容易了，比如Python、Shell、AppleScript、Ruby等，根据自己常用的操作系统选一个就好，如果你用Mac，这些语言可以任选，而且环境都是现成的，如果你用其他操作系统……，对不起，MacTalk只说Mac。这些语言除了能够帮助你锻炼意志和提高逻辑思维能力的同时，还可以在某些关键时刻帮助你处理各种繁琐复杂的工作，比如大量文本、定时任务、自动化任务、编写常用小工具等等，还可以引发跨界编程的轰动效应，不信的话去百度搜索“Python女神”便知。

好的，写到这如果还没有打消你学习编程的热情，那就可以继续往下读，下面才是正文：

要有光

无论学习什么，一定要有明确的目的和目标，如果是抱着玩票的心态，最多能够“知道”而不是“学以致用”，所以搞清楚自己为什么要学习编程，准备学习哪门语言，要达到什么程度，想用多长时间等等，这些问题在你的头脑里有个大概的思路和计划，就基本解决了Why和What的问题，下面我们来找How。

多说一句，其实学什么都有用的，大部分时间你只是不知道会在什么时候什么地方用。

经典教程

选定了语言不要着急去网上搜索各种秘籍、评价和下载各类盗版电子书，每个技术领域都会有一些经典的圣经级别的图书，找到它们，购买一本纸质书或电子书，最好是带练习题的，可以边学边做。

如何找到这些图书，豆瓣读书网应该是个不错的选择，虽然豆瓣的电影评价过于小清新口味，但图书评价还是值得信赖的，另外找乐于分享的老鸟推荐一下也是个不错的选择。

掌握基础，持续练习

每一门编程语言的学习内容都会涉及：基础运行环境、数据类型（数字、字符串、数组、集合、字典等）、表达式、函数、流程控制、类、方法等等，不同的语言还有一些不同的特性，这些内容并不复杂，尽快通过大量的练习击倒它们，然后再去深入了解面向对象、并发、异常、文件与目录、网络、标准库等内容，并辅以持续的练习，这些内容才能够让你真正进入编程领域并做出实际的软件。

初学者每天花1-2个小时是需要的，尽量保证阅读和练习的持续性和时间长度。其实1-2个小时根本不算什么，想想你们花费在看电视和刷微博上的时间吧，如果还说没时间，那就是不抽不舒服斯基了。

记住那句话：一边憎恶虚荣，一边找各种机会虚荣，在应该为了虚荣而努力的时候，丫拖延症犯了。

外事不决问Google

现代人的生活和学习是如此的方便，因为我们有Google！俗话说内事不决问百度，外事不决问Google，技术绝对属于外事，你要是去问度娘技术问题，被人家的回旋踢踢飞可别怪我没告诉过你。

以前学习技术只能通过技术图书和口口相传，现在遇到问题从Google那里就可以找到答案，所以用好Google你就能如猛虎加之羽翼而翱翔四海。如果你还在认为Google就是个搜索框，那就图样图森破了，Mac君今天为你推荐这两篇文章：

“Google，Google，再Google”http://wordpress.lixiaolai.com/archives/7572.html

“如何用好Google搜索引擎”http://www.zhihu.com/question/20161362

让你的搜索与众不同。

用好工具

俗话说的好，欲练神功挥剑自宫，sorry不是这句，工欲善其事必先利其器，想要学习编程一定要写代码，我们不提倡咬破手指写bloody code，所以一定要找到趁手的武器。我个人把工具分为三种，第一种是部分程序语言自带的shell，第二种是文本编辑器，第三种是集成开发环境（IDE）。

1、Shell，如果你在学习Python，那么python shell，bpython和ipython都是不错的选择；如果你在学习Ruby，那么irb就是ruby的shell；如果你在学习Shell，打开终端（Terminal）就是shell；如果你在学习Java或Objective-C，对不起，这些语言没有shell。

Shell能够单步执行你的编程语句并给出即时反馈，这种交互式编程方式非常适合初学者，所见即所得，所以凡是提供shell工具的语言，推荐大家优先使用shell学习。

2、文本编辑器，这个领域向来是“猿家必争之地”，溢美之词和吐槽之声交相辉映，从古至今绵延不绝，说起来都是眼泪，比如Emacs和Vim程序猿，大家沿着不同的道路和目标前进，但总是会在某个点交叉相遇，见面就扔石头和臭鸡蛋，砸得对方鼻青脸肿，然后擦擦眼泪和口水继续前行。还有IDEer说Vimer装逼，Emacser说IDEer垃圾balabala……种种血淋淋的事实足以拍一部惊悚科幻动作言情片。

我自己比较喜欢文本编辑器，但是也不排斥IDE，这种人俗称两边不待见，但我还是那句话，不为自己设限，不同的环境应该选择最好的工具。下面给大家推荐几款文本编辑器：

（1）VIM：号称编辑器之神，全键盘操作，充满速度感，良好的插件体系，几乎满足一切程序语言的编写需求。

（2）Emacs：神的编辑器，捆绑了文本编辑器的操作系统。没了，大家感受一下……

（3）TextMate：Mac专有编辑器，号称Ruby程序员最爱，当年1.0版一份39欧元，总共卖了十几万份拷贝，现在2.0免费开源，原来的开发者已经消失无踪，据说挣足银子去太平洋的小岛晒太阳去裊。

（4）Sublime Text：文本编辑器的后起之秀，发展迅猛，媲美TextMate，跨平台，比Vim和Emacs容易上手，号称性感编辑器。

以上四款自成体系，都有完善的插件生态环境，诸君可任意选择。

对于TextMate开发者赚了钱就跑的恶劣行径，大家完全可以批判，有时我们不得不痛苦的承认，国外程序员的鸡贼是我泱泱大国之IT民工

永远无法理解滴“泪”。

3、集成开发环境（IDE）

IDE是图形化的集成开发工具，具备精准的词法分析、编程提示、调试等功能，功能之繁复用户自知，如果做工业级编程和团队协作的话，还是推荐使用IDE。

在这里推荐几个系列：

（1）Eclipse系列，通过插件方式几乎支持所有的常用编程语言，免费。

（2）JetBrains系列，产品线丰富，几乎都是精品，Java、Python、Ruby、Php、Objective-C、Web等一应俱全，收费。

（3）Xcode，Mac上优秀的集成开发工具，所有的Mac App和iOS App都出自此货之手，免费。

微软的技术不懂，就不推荐了，嘿嘿……

除了写代码的工具，你还需要记录、阅读和查询，所以再为大家推荐三款应用：Evernote（笔记）、Pocket（以后读）和Dash（代码检索）。具体介绍和用法就不说了，不要忘了上一篇提到的Google君。

找到你的**Master**

小时候看西游记发现，师傅原来是用来人肉的；后来看天龙八部发现，牛人都不需要师傅，即使有也是要被别人一掌震飞的；再后来看射雕英雄传发现，愚钝的人首先得有师傅，其次得有很多师傅，再次每增加一个师傅功力都以指数级别增长，2、4、8、16……

所以，如果有人告诉你三人行，一个老师都没有，你至少要质疑这一观点，同时考虑自己会不会筋斗云，是否天赋异禀以一当百等等。如果不成，那还是去找师傅好了。

有老师的好处有这么几个：

（1）老师能够看到你自己看不到的地方，人这一辈子，很少人能给自己一个清晰的评价和认知，要么高估自己，要么低估自己，而旁观者，尤其是老师，往往能够看到你的弱点、长处、威胁、变化，并给你适时的提醒和指导，少走弯路。

（2）所有领域的知识都是成体系的，如果有这个领域的行家里手在你早期的学习阶段进行指导甚至设计练习技巧，与自己琢磨的效果是不可同日而语的。估计每个人都会有这样的经历，一个问题自己，想到心碎想到梦醒也没有结果，别人过来抽丝剥茧条理清晰的一讲，不仅你懂了，连你的小伙伴都懂了。这就是听君一席话胜读十年书的道理。

（3）好处多多，余不一一。

但是走出校门之后再想找传统意义的师傅就很难了，像绝地武士那样和Master出双入对同生共死更无可能，这时你就需要把身边的朋友、同事当做老师和资源，不耻下问，而且要问的有智慧，让人有回答欲望，那么如何提问呢，请参考我之前写的一篇“如何提问”。

参与社区和技术会议

自己学习和同事交流之余，可以参与一些网络社区的交流，推荐：

技术问答社区：http://stackoverflow.com，在技术领域几乎包括万象，无所不知。GitHub：https://github.com，几乎全世界优秀的开源软件作品都在上面。

另外还可以参与一些群组，订阅一些优秀的个人博客，这个时代依然有人愿意贡献优质内容。

选择性参与一些技术会议，比如QCon，不指望在会场能学到什么，但可以了解技术趋势，并看看别人在做什么。

刻意练习

之前写过两篇“刻意练习”的文章，自感对学习编程有一定帮助，大家可以去读一下。

逃离舒适区

这一部分适合已经有一定编程基础的童靴。

什么是舒适区？如果你是个新手，你就没什么舒适区，什么都不懂嘛舒适个毛，在磕磕绊绊的学习中懵懂前行，期间可能还伴随着老鸟的嘲笑和进度的压力，终于有一天你武功精进，乾坤大挪移练到了第五重，工作中开始得心应手游刃有余，不断有新人或老人来找你解决问题，你微笑着迎接挑战，淡淡的送走难题，你挥一挥手，不带走一片云彩，这是什么境界？这就是你的舒适区，这和靠在沙发上看电视的舒适不是一回事，通常进入舒适区需要花费你很多的时间和精力，需要你不断的练习，一旦进入，你会enjoy it！

这时候，如果有人胆敢让你脱离舒适区，可算要了亲命了，你会勃然大怒，轻则争吵，重则离职。这种事遇到太多了，一个写前端的你让他学习一些后端技术，一个写Java的你让他学习一下C，得到的答复可

能会，Sorry, I feel very uncomfortable!

没有人学新东西的时候非常舒服，一旦经历过从新人到老鸟的过程，再让你进入陌生的领域，那种痛苦会让你自发的去抗拒。但是一个人不可能永远躲在舒适区里，逃离舒适区会有助于你从不同的角度看问题，视野会更加开阔。人总要往前走的。

很多人在某个地方待久了就会非常懈怠，没退休就像在养老，这时候你就知道，他们在舒适区太久了，与在哪个地方无关。

最后一招“见龙在田”

实战总是很重要，为大家推荐一个在线学习编程网站：

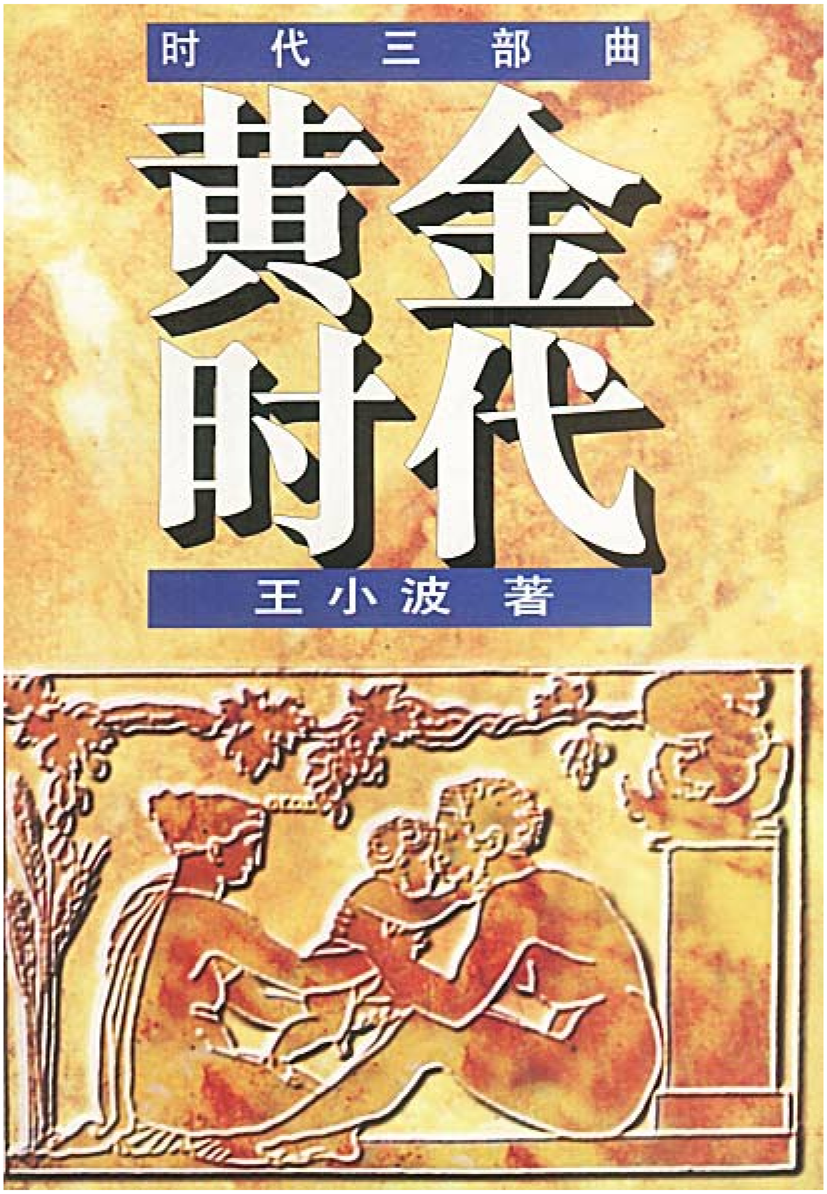
假以时日，各位必定武功大成，那时横刀立马、拔剑四顾，说英雄谁是英雄！

神奇的程序员——王小波

【发布日期　2013年4月12日】

喜欢读书的人，我想对王小波都不陌生，他是中国最富创造性的作家之一，他是中国近半世纪的苦难和荒谬所结晶出来的天才，他英年早逝。他的作品对我们生活中所有的荒谬和苦难作出最彻底的反讽刺。他还做了从来没有人想做和做也没才力做到的事：他唾弃中国现代文学那种“软”以及伤感和谄媚的传统，而秉承罗素、伊塔洛·卡尔维诺他们的批判、思考的精神，同时把这个传统和中国古代小说的游戏精神作了一个创造性的衔接。（部分内容摘自百度百科）

我读过王小波三部曲黄金时代，白银时代和青铜时代，读过沉默的大多数，读过红拂夜奔，读过似水流年，但是最近才知道，原来王小波是一个伪装成作家的程序员（这位同学不用站起来了，不知道这是调侃么）。



* 王小波很早就接触了计算机，1988年毕业于美国匹兹堡大学东亚研究中心，获硕士学位，那时就知道Macintosh，玩过IBMPS/2。

* 90年学习FORTRAN，开始进行数据统计，并开始在北大教统计学。

* 91年搞了个用调整字模发生器方法输出汉字的系统，并发出嗟叹：嗟夫，不过现在我对微机已无兴趣，因为发现写小说也可赚到钱。一代天才作家终于开始写小说了，但他依然没有停止编程的脚步。

* 92年开始写C，“用C编的软件已经用熟，并做出了各种写小说的工具，别人的软件已不用了。现在主要是写书赚钱。从今年初开始写长篇，首先做了写长篇的专用软件，现在基本调通，开始写了”。

* 93年开始搞汇编，“下决心买了一台286，这些日子在改造软件，作了不少汇编工作。其核心是它在虚拟保护方式（virtual address protected）下工作，以便利扩展内存（expandedmemory）”。

* ……

如果王小波不那么早辞世，估计会有更多伟大的文学作品和软件产品留下。谁说文科生不懂技术？谁说程序员不能写书？这才是技术与人文的完美结合啊……

点击原文，查看王小波的软件历程。

MacTalk



Mac上的软件付费

【发布日期　2013年4月13日】

关于这个问题，有很多位读者曾经问过，为什么在Windows上软件都是免费的，到了Mac都要付费才能用呢？所以就决定写写这个话题，曾经在微博上看到一位父亲发的内容：

> 在我儿子帮助下把iPad越狱装了个PP助手，看着那么多不要钱的软件想着以前竟然花钱买软件简直是痛不欲生啊，不越狱的苹果简直就是一颗生苹果啊！

我的感受：这儿子也够可怜的，从小就不知道偷盗为何物！

好吧，今天就说几句软件付费的问题。

以前有朋友看我用Pixelmator处理图片，问我从哪下载的，我说从App Store上啊，付费软件。他说啊，你们Mac什么都要钱，你看Windows上的PhotoShop都是免费的。

当时我就崩溃了，PS软件无论是在Win上还是Mac上，都是相对较贵的个人软件，只是国内盗版严重，居然很多人不知道这是付费软件。嗟夫杯具！

Windwos操作系统和Office软件的盗版流行，我个人以为和微软的定价策略和垄断政策还是有一定关系的，十年前一套Windows和Office和现在的价格差不多，对于当时的工薪阶层还是有压力的，但是盗版为微软形成了全面的垄断市场，反正微软在中国市场钱也没少挣，人家对于这点损失承受得起，毕竟还有广阔的国际市场。

国内的个人软件开发厂商就惨透了，盗版毁掉了我们的通用软件整整十年，早期做通用软件的公司要么倒掉，要么转行做企业软件和互联网。我在洪恩软件的时候，我们研发的个人教育软件，在整个市场上只有10％或者更低份额是正版，其他全部是盗版。大量做通用软件的公司死掉，金山这种大型软件公司存活并发展，但是也并不是靠通用软件盈利。

在这种环境下，国内盗版商笑眯眯的做盗版，国内用户乐呵呵的用盗版，殊不知摧毁的是民族软件工业。

反观国外，如果一个程序员开发了一款不错的个人软件，基本上就可以衣食无忧了，比如Mac上的TextMate1.0，总共卖了10几万份拷贝，一份39欧元，大家算算这个程序员的收入，最终他把TextMate 2开源了，因为不需要卖钱了嘛，而且这兄弟成为富翁以后基本上不再对TextMate改进了，结果开源后，大量爱好者开始加入开发，新特性不断增加，最终形成了良性的生态圈，目前发展得非常好。

我个人也用过盗版，但我始终知道这是不对的，说白了和偷东西没有太大区别。转到Mac上之后，发现Mac上也有大量的免费软件可用，另外收费软件也完全可以承受，操作系统才100人民币多，类似Office的Pages／Keynote／Numbers可以分别购买（128），其他很多个人软件都是几块钱／十几块／几十块，只有一些很少用到的专业软件才成百上千，所以我在用Mac之后，逐渐变成了正版用户，我想这也是大量的普通用户和程序员开始转向使用Mac的一部分原因吧。

关于软件盗版和付费，我的看法是：

1.盗版肯定是不对的，如果用了盗版软件，至少要有愧疚之心。如果你是个穷学生，学习软件开发用了盗版软件，谁忍心责备你呢？但大家千万不能无耻到开篇提到的那位父亲那样，不仅误己，而且误人。

2.程序员也是要吃饭的，你们每个人在自己的电脑上使用的每个软件都是程序员一行行的代码敲出来的。

3.在经济实力允许的基础上，尽可能用正版，尤其是程序员。程序员不支持程序员，还怎么指望别人呢？

不要做一个Hater

【发布日期　2013年4月17日】

MacTalk开通以后，我收到过很多寻求建议的问题，也尝试回复过一些，还有一些是我没有能力回复的。人生一路走来我们会寻求很多建议，也有很多人给你忠告，需要警惕的是，这里面有相当一部分人的“忠告”总是负面的，比如你想去学编程，他说，你的逻辑能力不适合编程，你说要去做销售；他说，性格决定命运，你的性格做不了销售；你说我要去创业，他说，这个项目类型没人会投资的，早做早死晚做晚死；你说我要站着把钱挣了，他说，这是在中国……当你稍微遭遇了一点失败的时候，这些人就会祭出万试万灵杀手锏：你看，我早就说过……



我们把这样的人统称为Hater，这种人对自己不了解或没有勇气尝试的事物永远持否定态度，如果你发现一个人大部分时间在否定着什么，那么他们的意见不听也罢，甚至于那些鼓励的建议也仅仅是建议而已，仅供参考，因为最终不是那些提建议的人去做事和承担后果。做任何事情都是我们自己的选择，想清楚了也好，没想清楚也罢，想去做的话，尽可能鼓励自己去做，做实事的人总是让人敬佩的，而且由于我们在做事，所以总会遇到失败，这时候那些口诵大悲咒“我早就说过”的

Hater是完全可以忽视的，因为所有人都是在试错中成长，那些不犯错的人充满了各种幻觉，其实是因为他不再成长了。

李笑来在《把时间当做朋友》一书中写道：他们一定要给你泼冷水的。泼冷水的愿望之强烈，你无法想象。那种强烈借助了太多的力量：怀疑、嫉妒、恐惧、愤怒。而在表现的过程中却又包装上另外一层表皮：关怀、爱护、友爱、帮助。

当然李笑来没有把“他们”定义为Hater，但我想应该是一个意思。所以我们首先不要自己成为Hater，另外也没必要去听取Hater的忠告。地球也不会因为Hater的存在而停止转动！

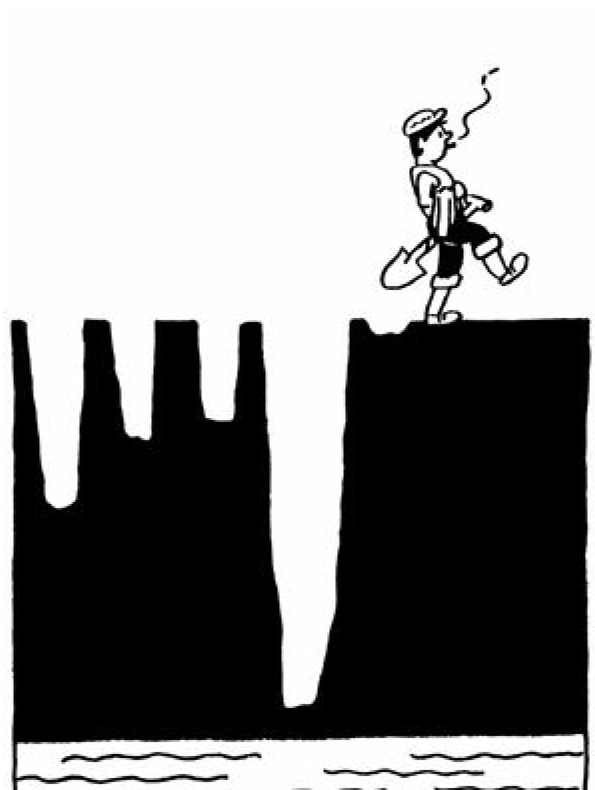
当老罗的锤子手机发布的时候，我仿佛是看到了一群Hater冲上去一顿狂贬，当然里面也包含了一些看似有道理的贬，但是这些东西有什么力量呢？老罗不也说了么，你们的感受我根本不在乎。我不认为锤子有一天能够砸烂苹果，但锤子会有自己要砸的东西……

当时就有写这一篇的冲动，不过一直放到今天才写，是为记：Don't be a Hater!

沉默的坚持和沉没的成本

【发布日期　2013年7月8日】

快速解决掉晚饭之后，我跳上坐骑，快马轻裘赶往机场（不是说不让养马么？），今日帝都天气极佳，奔驰在高架桥上可以看到远远的山脊，山顶覆盖着层层波浪也似的云彩，将落未落的夕阳把最后的光晖洒落在山顶和云层，紫霞满天，让人产生一种这就不是北京的错觉！一路顺风，耳边听着谢天笑、鲍家街43号、左小、声音碎片、超级市场的歌，很快就抵达三号航站楼，停好车蹿到航班接机的B口，一打听，好吧，早来了两个小时。



找了一个蚊子多的地方（这当然是后来才发现的）坐下，左右看看大家都在玩手机，要么刷微博要么玩游戏，做为一个有志中年，我决定把手机放进口袋，拿出iPad mini开始阅读一直没怎么看的“推荐系统实践”，一口气读了一个小时，直到蚊子开始提醒我，该歇歇眼了，我发

现胳膊腿上有不少提醒的痕迹，于是我也觉得该歇歇了，当然另一个原因是看到算法部分实在看不下去了。

于是我换了一个蚊子少的地方，打开第二本“清醒思考的艺术”，开始读第五节“纠缠于沉没的成本”，感觉挺有意思，比算法好玩多了。文章开篇就举了个例子，丈夫陪妻子看电影，比如电影叫“中时代”，看到一半丈夫实在看不下去了，说，“亲，咱回家吧”。妻子不同意，“钱都花了，别说中时代，就是小时代都特么得看完”！这就是沉没成本！不管是谈恋爱、做项目、创业、投资都会面临这沉没成本的问题，基本的句式就是『我们已经投入了这么多……』『我们已经走了这么远……』『我们已经牺牲了这么多……』。纠结于过去的沉没成本，会让人感到痛苦和犹豫不决，该放手时要放手，大概就是这章要表述的思想。但是作为一个拧巴的程序员，Mac君有话要说。

我个人曾经参与过很多失败的项目和产品，其中大部分是在研发和经营过程中被叫停的，也就是不纠缠于沉没的成本，这也许是对的，但有几个项目我至今耿耿于怀，我不知道坚持下去是否有成功的可能，毕竟后来同类型的东西有坚持下来的成功者。也就是说，很多时候你并不知道那些沉没的成本是否真正沉没了，你只能沉默的坚持，希望坚持到最后的水落石出。

我曾经写过一句话：你一直坚持着最后失败了，就是固执不懂变通。中途转向失败了，就是没有再坚持最后一公里。相反，如果你一直坚持着最后成事了，那就是无所畏惧一往无前；转向成功了，那就是灵活柔性随需应变。可见选择是一件多么艰难而奇妙的事。是沉没的成本还是沉默的坚持，都是你自己的判断和选择，有时候是命数，有时候是形式，只要是自己选择，接受就好了，只能这样。

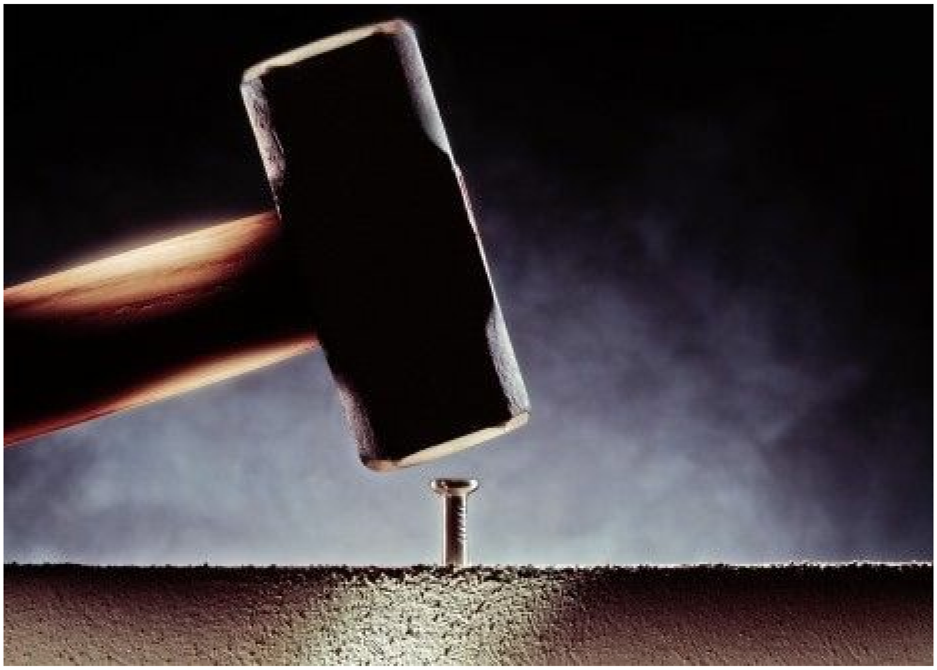
当Mac君思绪万千的时候，两位领导的座机已经稳稳的降落到了首都机场，看到迎面而来的笑容，我感觉书没白读，上车回家！

再说一句，有条件有时间的时候，尽可能去机场车站接接家人朋友，那种笑容确实非常让人开心！

锤子和钉子

【发布日期　2013年7月9日】

曾经有位古人说过，如果你手里有一把锤子，所有东西看上去都像钉子。还有一位今人说过，如果你有一个钉子，就会满大街找锤子！



哪位读者说说这话是什么意思？好，就这位穿黄衣服的童靴，请把手里的锤子放下到前面来“……此处省略一万字”。

这位童靴讲的非常好，不过我要纠正两点，第一，诺基亚不是锤子，虽然少数型号可以当锤子用。第二，锤子不是老罗手机，虽然他的手机叫锤子。

如果你们以为我今天要讲的是手里有锤心中无锤不要见人就锤，工具是死的人是活的，不要被工具禁锢balabala……你们就图样图森破了，Mac君今天要讲的是：

1、尽可能去做开发工具的人

无论是锤子还是钉子，都是工具，如果让我来选择，是去做锤子和钉子本身，还是去用锤子把钉子敲进木板，我显然会选择前者。一个团

队里最容易被低估的就是做锤子的人，尤其是做内部用的锤子，因为他们并不直接创造利润，无论做出来的是雷神之锤还是天马流星锤，只要不是作为产品直接销售，那么就是公司买单，而那些拿着工具到处晃的人则更容易出彩，因为客户买单。

从做工具和使用工具这两件事的认知程度上来说，前者更容易知其所以然。从长远来看，个人以为工具和平台是推动人类发展的原动力，业务和应用则构建于工具和平台之上，相辅相成。

以前我说过一句："阿里玩的是平台，顺道把电商做了；京东玩的就是电商，顺手搞搞技术"大概表达的也是这个意思（当然京东现在也开始做平台了）。你以为阿里在做电商，其实人家在做电商平台，你以为苹果在做电脑和手机，其实人家在打造App平台。

有了做工具的人和工具，才有可能实现几十倍甚至成千上百倍效率的提升，所以，有机会的时候，去做工具吧。

2、用好工具

写到这估计有同学开始愤愤不平了，不是说不要去造轮子吗？注意，人家说的是不要重新发明轮子，不是不要造轮子！如果有了好的工具和平台，那就去用好了，不要重写一个，尽可能的复用，而不是犯NIH综合症（Not Invented Here，就是非我所创、不是爷搞的就不用的意思）。

用工具，而且要用好工具。用好工具的基础是对工具有足够的了解并刻意练习。很多人从Windows转到Mac上没用两下就开始抱怨了：天塌了地陷了，开始菜单不见了！妈妈开始菜单不见了我该怎么开始？桌面图标在右边怎么办？键盘变了怎么办？找不到文件怎么办？

对于这样的使用者，首先用一记360°回旋踢让其冷静下来，然后再告诉他，你知道Dock吗，了解Finder吗，会用Mission Control吗，那么宽的触摸板不是让你烤红薯的好吧，找文件要用Spotlight，神器Alfred一定要装上，什么都不了解您拿着鼠标里三层外三层的戳能不累吗？

我见过最神奇的Mac用户是把Spotlight关掉了，问之何为？答曰因为每次建索引太慢臭。仔细一问才知道，此仁兄从来就没有完成过一次完整的索引，每次装系统建索引的时候感觉费时费力就把Spotlight关掉了，真实的情况是Spotlight只有第一次全索引慢点，只要建完之后就是增量索引，用户基本无感的。这种用户我们只能说图样图欠揍了。

后来我告诉了他解决方案后，他连声惊呼，太神奇了，太神奇了。好吧，他在Spotlight关掉的情况下用了一年Mac！

好，总结一下，今天的话题是去做工具，并用好工具。

读书日谈书

【发布日期　2013年4月23日】

1995年，联合国教科文组织正式确定每年4月23日为“世界图书与版权日”，设立目的是推动更多的人阅读和写作。我觉得我正在践行这一点，今天是读书日，就谈谈书的事。

工作十几年来，我发现书是自己购买频度最高的商品，基本上看见好书就买，或者临时急用的工具书也买，甚至碰到近期不会读的书，觉得不错也就买了。工作前期，薪水低微，那时候买书要考虑价格，后来相对好一些，价格已经不再是买书的考虑因素，买书的地点也从实体店转移到了网络。现在算一下，还真想不起来哪本书是在书店买的，或有多久没去实体店买书了。

之所以爱买书，一方面是因为喜欢看书，另一方面也和自己从事软件行业有关。软件行业对人的要求是终身学习，技术的革新和变化太快，2000年的软件技术和今天相比，完全不是一个量级的。另外还有第三个因素，也是我很长时间没有意识到的，那就是书的价格。2000年书是几十块钱一本，到现在基本上还是这个价位。而2002年4千元一平米的房子，到了2012年已经变成4万一平了。即使是从整个时代变迁的角度，书也算是涨价最慢的商品之一了，所以，买书是个事儿吗？

既然买书不是个事儿，那为什么写书的、翻译书的、出版社的，都没有香车宝马夜韶明装呢？很多写书出书的都有其他工作，完全靠写字养活自己的，国内就那么几位，号称畅销书作家。青年作家兼赛车手韩寒在《给李彦宏先生的一封信》中写到，“一本书如果卖两万本，已经算是畅销，一个作家两年能写一本，一本可以赚三万四，一年赚一万七，如果他光写书，他得不吃不喝写一百年才够在大城市的城郊买套像样的两居室。”之前与一位资深出版人聊天，他说好卖的技术书籍的标准是5000册以上，10000册以上已经是畅销书了，即使这样，出版社和作者、译者的利润也非常薄，这一点还是很让人震惊的。在了解这些背景之前，我琢磨着一本畅销技术书籍，怎么也得卖到十几万册吧？中国可是有十几亿人口基数的，但现实总能轻易突破你的底线。

什么原因造成这样一种局面呢？我个人觉得有三点可以探讨：

1、现代人普遍不读书

以前看见一个段子，说：“在日本地铁里，5个人就有5个人读书看报；在台湾，5个人就有3个人读书看报；在香港，5个人中有2个人读书看报；而在中国的地铁里，5个人中往往有两个人在讲话，另外3个人在听他们讲话。”当然，现在北京地铁里都是人贴人肉贴肉，每次挤地铁都祈祷自己立刻瘦十斤，能说句话就不错了，哪还有空间读书呢？不过这从侧面也反映了一些问题，不管我们承认不承认，有相当大一部分中国人，成年后就不读书了。既然书都不读了，买书就更少了。事实上由于技术从业者的强迫性学习，搞爱踢的已经是一个非常爱买书的圈子了。

2、国内对知识产权的漠视

对知识产权的漠视，这是我们永远无法忽视的问题，尤其是在图书音像和软件领域。软件就不说了，之前的讲过一期。对于图书，无论是纸质书还是各类电子书，盗版界基本采取了广泛撒网、重点捕捞的战略战术，做到了宁可错盗一千，绝不放过一本的辉煌战果！如果一个民族不尊重产生和创造知识的群体，那未来我们还能留下什么？

3、电子书对传统产业的冲击

我现在阅读基本是三种介质：纸质书、iPad mini和Kindle。这些优秀的电子阅读工具极大地冲击了出版行业，对于书价提升空间很小的作者和出版商来说，打击是超乎想象的。

电子书之所以能对纸质媒体产生这么大的影响，主要原因是软硬件技术的进步。现在的iPad系列和Kindle系列，都是很好的阅读工具，尽管阅读只是iPad的一个功能，但是我知道很多人用到iPad最多的功能就是阅读。由于这些电子版本的书籍，去除了大量的生产成本，书的价格可以降到很低，比如Kindle Store上的很多电子版书籍只有纸质书籍的1/3，国内的多看、字节社和豆瓣阅读上的书价都非常诱人，而且这些云端的书店还提供了大量的免费书籍吸引读者，优势显而易见。

所以对于书、阅读和出版行业来说，未来的方向只能是电子化。

纸质书不会消亡，总有和我一样的人喜欢手捧纸质图书阅读的感觉，对于纸质图书和传统出版业来说，除了提供更好的内容和更好的质量之外，还需考虑的就是图书的电子化。

传统出版业和电子化图书的结合，前景更为广阔。电子化并不是单独做成PDF或一本书一个App放到移动终端就完了，而是要把云端的书

城和移动的终端结合起来，这样用户就会在阅读的过程中获得更多的好处，比如笔记保存、分享、高亮，进度同步等等。这一点国外市场已经做的非常成熟了，在苹果的iBook Store或亚马逊的Kindle Store中几乎可以找到你想买的任何书籍，而且除了良好的阅读体验，这些阅读器都对应了强大的后端系统，iPad有iBook Store，Kindle有Kindle Store。而国内的类似应用云中书城、多看、字节社、豆瓣阅读等，书库的规模相对而言还是比较小。如果传统出版业和这些电子厂商能够很好地结合，那么无论是对于书商、云端书店和读者，都是非常好的事情。

当然，图灵公司已经在做这些事情了，大家已经可以在多看和豆瓣阅读上可以找到很多图灵策划的书籍。

在阅读App上，国外首推苹果的iBooks和亚马逊的Kindle，基本上都是简洁优雅大巧若拙的风格。国内首推多看，字节社、豆瓣阅读、云中书城也不错。不知道我的读者里是否有豆瓣的童靴，我特别奇怪豆瓣阅读的iPad版本，为什么不增加调整字体大小的功能？用iPad时还不觉得，字体大小正合适，但是在iPad mini上字体就太小了，还不能调整，而iPhone版和Android版都是可以调整的，这个问题着实让人费解。

总体来说，我希望大家读书的环境越来越好，能读到更多的好书，也希望提供创造性内容的工作者能够得到应有的尊重，包括精神的和物质的。也许不远的未来，创新和创造会越来越多，复制和盗版会越来越少！

付费阅读

【发布日期　2013年5月19日】

之前做过一次付费阅读调查，仅仅是个探讨，因为微信公众平台的作者无法确定收费规则，最多搞一个支付宝的donate，自愿捐助，这也是目前很多作者采用的方式。

调查结果也很有意思，简单的看了一下，20%的反馈者属于铁杆读者，认为文章定价太低，“好的信息，一条10元都值得”“必须订阅”。还有20%的游离态读者，“党费一年才两块四呢”“如果付费的话则会取消”。剩下的60%基本上属于中坚读者，表示每月1块可以接受。当然也有表示每年10块可以，每月1块不能接受的。这也是我特意想求证的一点，每月1块钱和全年10块，其实差距只有每年2元钱，我想知道时间走到2013年这个号称大数据的时代，是否还有人在意这2元/年，事实证明，有的，而且不是个案。所以你就知道，永远不要把自己的认知想当然的代入到消费者和用户的想法中。

当然还有更多的读者没有任何反馈，他们属于沉默的大多数，这个时代，这样的人，总是最多的。

关于软件、书和文章的付费问题，我个人有两个观点，第一，首先要关注我们提供的东西是否能够真正解决用户的问题、满足需求并带来价值；第二，付费总是能够在某个阶段为作者带来更旺盛的创作热情，在某些时候也可能适得其反。

很早我就认识到的一件事情就是，人们乐于去花钱买好的东西“当然我指的的普通人，不是强盗”。从长远来看，人类对美好的事物的追求是永无止境的。所以不要担心他们是否愿意买你的服务，不要担心他们是否会把钱花在你做的东西上。如果有人不愿意并抱怨你做的东西，这没什么，因为这并不是他们真正需要的东西，仅此而已。

世界上还有大量的人懂得欣赏，愿意付费去购买美好的东西，因为他们认为它是值得的，它让他们的生活更美好，只要你的东西足够好。我想这是技术人持续编程和写作的最重要的理由之一！

另外，就我个人的软件使用体验来说，收费软件的平均水平远远高于免费软件，除非这个免费软件的背后有强大的社区和公司支持，比如优秀的开源免费IDE工具Eclipse，就是由IBM支持的，可以说Eclipse是世界上程序员使用最多的开发工具。但即使是这样，与Jetbrains的付费

软件Intellij IDEA相比，Eclipse的功能依然稍逊一筹。没有人能够饿着肚子写出优秀的作品。捐助和付费，从长远来看，能够获得更好的回报。当然，也可能出现的情况是，创业者或创作者由于用户和读者的付费发财了，慵懒了，把公司卖掉了，或停止创作了，这会导致产品的质量下降或停止更新。但是从另一个角度来说，这是这些创业和创造者应得的。因为大部分人在朝九晚五和电视电脑前消磨人生的时候，他们则在孤独的环境中默默的创造内容。

没人欠用户什么，大家都有选择自己生活的权利。像乔布斯那样坐拥百亿资产身患绝症依然充满创作热情的，这个星球上就那么一位。比尔·盖茨做不到，保罗·艾伦做不到，史蒂夫·沃兹也做不到。

付费和免费，这是个问题，但不是我的问题。

技术成长

【发布日期　2013年5月13日】

越来越多的技术人员最终走向了领导岗位，也有更多的技术人员在技术领域走的更深更远，孰优孰劣如何选择？

是否要一直留在自己熟知的领域不断深入，还是降低技术创新和跟踪来加强交流、激励、演讲、组织、经营和财务能力？



在抛砖之前，我觉得可以先引玉，看看大家发来的反馈：

『大路』的看法：我觉得首先要认识自己是什么类型的人。假如更享受和机器、代码打交道，喜欢比较直接和单纯的人际关系，那做一个不断深入的技术专家很不错。如果较为擅长沟通交流，享受通过整合更多人力量系统性解决问题的成就感，那就在保持技术敏感性和理解力的前提下尝试走向管理，也许是不错的方式。

『广杰』的专业：从两方面分析比较靠谱。一方面是自己的需求在马斯洛第几个层次上，需求是什么，最后找到自己究竟想要什么。另一方面，认真分析自己的特点，例如可使用九型人格着重分析。自己先实践，后找keyman咨询，找伙伴一起评估分析自己特点。最后才是看看

圈子或者三层关系网，资源人脉情况。哪个方向只要自己真的喜欢，肯投入，锤炼出的价值一样ok。

『王力拓』的叹息：每个人都沉下心搞一辈子技术？在这个浮躁的社会氛围里恐怕不现实

『托狮卡尼尼』的选择：貌似更多的人会选择领导。更有话语权，钱可能也更多。但是哪个更让自己快乐就难说了。

『TimonWang』的困惑：喜欢技术，但是已经莫名其妙的走上了管理道路，虽然也还开发，但是各种方案文档、各种项目内部会议层出不穷，占用了大量的时间。技术呢，大方向上的技术研究研究，技术细节关注少很多了，可能和自己在技术钻研上还不够深入也有关系。

『傲刀客风』的结论：必然有一部分技术人员需要走向领导岗位，因为有着技术背景，所以更容易为团队指引方向。但是大多数的技术人员需要向更深处努力，毕竟技术才是王道，空谈而不技术实现，产品怎么进步？如果你的技术比不过大多数人，那还是努力锻炼眼光吧。

『Brother』蓝天的判断：取决于每个人对自己欲望，价值观，成就来源，痛苦承受力等方面的判断！

『刘翔』的问题：我也很想知道那些终未走上管理职位的技术人员后来都去干嘛了？

最后还有一个比较专业的回复，内容较多，来自『爱小恩』

关于是走管理还是技术路线，我认为核心有两个因素：

1、个人的兴趣与性格导向

2、公司及大环境本身是否可提供持续学习与发展的技术或管理平台。

一个人如果极有兴趣在某个技术领悟钻研一生，但他热爱的这个公司甚至当前技术大环境无法提供其继续钻研技术的平台，很可能他逐步就转向了管理；如果一个热衷于交际与管理的人，即使技术科班出生，未来走向管理路线的概率也注定更大，反之亦然。

而究竟技术路线更好还是管理路线更棒，这个问题的答案也许是我们认为两条道路相同，但实际并非如此。

管理路线在人们潜意识中具有更高的地位和更让人信服的领导力。

举两个事例：

远的，《人月神话》曾用很大的篇幅讨论过相同级别的技术岗和领导岗应该具有相同的薪资、待遇，甚至细化到应该具有相同的办公室大小。说明什么问题？技术岗所受到的待遇普遍是低于相同层级的管理岗位的，技术人员渴望能够扶正。但即使做到，在人心中的地位感受也是有微妙差别的。

近的，这点也能说明为什么技术岗地位常常低于相同管理岗。NRM曾回答过“为什么项目经理拿的钱比程序员多”，因为，“善于交际的人稀少，这种人在任何公司都能来到巨大的好处，尤其是在软件开发领域”。对于公司高层，管理价值相比技术价值更加凸显。热门、先进或价值巨大的技术，注定会被人趋之若鹜，伴随的也是该技术报酬的降低，所以技术人员不得不持续学习，以提升甚至只是保持自己的价值。而真正高级管理层所需要的更是持之以恒及全方面能力的学习和培养，与人打交道以及管理人的哲学，对于很多人来说，更是难以学习和快速积累的。

以上内容均为读者提供，文字和标点上略做调整。珠玉在前，下面我再谈谈关于技术成长的一些自己的看法。

讲个小故事先：

村里有一个傻子，他的古董手表不走了，拆开手表一看，发现里面有一只死蟑螂，傻子恍然大悟，『怪不得手表不走了，原来管事的家伙完蛋了』。

不知道大家看完这个故事是怎么想的，我的想法是，永远不要简单的割裂的看问题，很多时候我们以为找到了问题的答案，真实的情况是，我们只触摸了表象，盲人摸象看起来是个笑话，但在现实世界里却随时随处的发生着。

落到技术成长这个话题上，那就是技术人员在成长的过程中，关于技术和管理（或其他非技术技能），永远没有非正即负、非黑即白。

在计算机发展史上，确实有很多技术天才醉心技术不问世事，比如大家比较熟悉的软硬件技术天才史蒂夫·沃兹，他虽然是苹果公司的创始人之一，在苹果初创的时候几乎以一己之力单枪匹马做出了AppleI和AppleII，对苹果公司的贡献居功至伟，但他对自己的定位却非常简单，就是去做一些酷的东西，成为一个伟大的工程师，写出匪夷所思的代码，并且跟每个人都成为朋友。

这样的人是不需要去演讲、去管理、去销售、去协调资源的，让他好好做技术就好了。

世上能达到这种级别的技术天才凤毛麟角，我们是这样的或接近这样的技术天才吗？

如果回答是肯定的，下面的内容就不用看了。如果是否定的，可以接着读读。

（一）先谈谈选择

大部分人进入IT行业是因为喜欢技术，还有一部分可能觉得程序员起点工资好一些，无论如何，进入技术领域之后，只要用心用功，大部分人会得心应手，在自己的岗位游刃有余。这时候不管是主动还是被动，都会有更多的责任找过来，你不再是搞定自己的任务就可以了，你需要去协调、交流、传播、聚合。每个人在这个阶段都是不适应的痛苦的，有些人挺挺就过去了，进入另一个更大的空间，有些人以不喜欢做繁琐的事务工作为由，继续回到原来的领域深入研究。

这两种选择，无所谓对错，但是有一点我想强调的是，不要轻易为自己设限，人的潜力是无法估量的，作为程序员，无论是哪种选择，都需要在专业的基础上尽可能的扩大自己的知识领域和综合技能，总是躺在技术舒适区是容易的，比如你是一个IBM大型机的开发人员，你可以在最后一台大型机陈列在博物馆之前不去看任何其他技术。但是一旦这一天来临，你要承受的压力是巨大的。

另外，作为技术人员，多读一些人文类的书籍，好处多多不费电！

（二）再谈谈领导。

首先我想说的是，如果大家看到领导和管理这些字眼，想到的仅仅是发号施令和挣更多的钱，那么就很难做好事情，这一点可能很多人不以为然，不急，大家慢慢体会。

技术领导在成为领导之前，一般是某个领域的技术明星，在成为领导之后，遇到最困难的事情就是没有那么多时间去接触和研究最新的技术，你会很尴尬的发现，有些技术问题你不得不向你的下属请教，这种成长的痛苦是非常折磨人的。

我个人就经历过类似的痛苦，不过现在我已经能够非常坦然的向团队成员请教问题了，因为技术领导需要的是敏锐的技术嗅觉和前瞻性，规划能力，解决问题的能力，组织协调的能力等等，而不是去掌握团队使用的所有技术。

而且，还需要注意的是，一个团队中除了有任命的领导，还有更多未任命的领导，他们往往充满活力，善于提出各种奇思妙想，自发带领团队以更有效率更优雅的方式工作，这样的潜在领导尤其值得关注，他们的发展空间可能更为广阔。

（三）最后说说白衣飘飘的年代

当有一天你终成一代大师，白衣飘飘风尘仆仆，知行合一物我两忘，那时你追求的是如何找到终极答案，而不是要这个答案一定是自己回答的，因为大师有足够的自信去驾驭那些人和物，最终形成可用的资源。

性格决定命运，选择决定领域。世界上只有极少数人能成为大师，这个人不是我，但我希望是你。

克隆高手

【发布日期　2013年6月17日】

不知道多少读者看过最近汤姆·克鲁斯的新作《遗落战境》，影片之宏大和剧情的逆转就不细说了，让我印象深刻的是外星势力通过克隆人类的精英分子来反制人类，掠夺资源的场景。阿汤哥在电影中扮演了一枚帅哥宇航员，要能力有能力，要Pose有姿势，力量与智慧的化身，沉着冷静，风采迷人，实属人类之精华，当然还有一个靓妹纸就不说了，这俩人在一次出航中被抓，外星人估计也是好色之徒，一看帅锅靓妹，直接克隆几千份，随用随克，一会派去东塔一会送到西塔，一会去修理几个机器人，一会去修理几个抵抗者，然后自己在摄像头后面笑的像你大爷！

各位看官看到这估计就明白了，这帮外星孙子就是慕容一脉复兴啊，这不就是以彼之道、还施彼身么，看来外星大爷也是一点不厚道。



但是大老板看到这估计就嗨皮了，我知道很多老板经常对公司里牛人说，“你看，你这么牛，但是你就一个人牛，这样不好，对吧？你就不能给我培养出xxx个你来，那样不就你好我好大家都好了么？”，这时候估计高手都会坚定的回复一句“你大爷！”

我记得当年京东促销导致服务器全面崩溃时，京东老刘放话“增加三倍服务器，活动再搞一次”，做技术的都知道，想做到性能容量随着服务器数量的增加线性扩展是多么困难的事，没有积累想通过复制绝无可能。机器如此，况高手乎？

所以高手除非能够克隆，每个靓汤都一样英明神武，有灵感有直觉，处理事务解决问题思路都一样一样的。

若非如此，高手不可培养，只能独自成长。

说到直觉，我觉得这是高手的一个特征。虽然自己不是高手，但有时候帮助别人解决问题时往往靠直觉行事，有时候还会觉得，这么简单的解决方法对方怎么想不到呢？这就是直觉吧。你有可能说不清楚为什么会想到这个方法，但是知道，那是正确的。我们家老太太有一天冬天患病手疼，我查了大量的医学资料，觉得有可能是腱鞘炎，随后带老太太去医院，那位医生让俺妈简单的做了几个动作，看了看手和臂，简单问了问，说，很可能是类风湿，最好做检查确认下。检查结果确诊为类风湿！这就是高手，他可以通过一些微妙的线索得出结果，而新手往往让你做一堆检查，最后得出一个错误结论！

直觉，这玩意能复制和培养么？

还有，很多公司往往希望通过设置规则和秩序来保证专家和高手有更大的产出，殊不知这样的做法正好适得其反。在技术公司，规则和秩序，更多适合那些不知进取或不思进取的新手和胜任者，人一旦在某个领域达到一定的高度，就有了自己的做法和直觉，方向没问题就行了，复杂的制度反而会抑制他们的创造性。另外，自信往往来自于无知而不是知识，一个人一旦成为高手，就会痛苦的意识到自己知道的是多么少，人生苦短学问太多，这样的人是不需要去鞭策的，给空间和鼓励就好。

扯了这么多，估计有读者要问，“你丫见过高手吗？”

我还算幸运，刚进入洪恩软件的时候身边有几个牛人，其中之一还是我的顶头上司，这哥们技术功能非常扎实，据说读汇编像读《读者》一样，公司要做网站，他就学了几天perl，然后带着我们把洪恩在线搭起来了，后来要做企业级软件，他又学了几天java，带着我们把产品做出来了，那是2001年左右……后来他去创业鸟。另一个是独行侠，大老板专门为他配了办公室，里面一堆服务器和显示屏，他经常蜷缩在宽大

的座位里抱着键盘编程，我们进去请教问题时看到屋里烟雾缭绕，丫叼着烟翘着二郎腿正在敲打键盘，后来这兄弟搭建了完美时空游戏的底层框架和引擎……。还有一位被大老板誉为国内少有的能够写操作系统级别代码的家伙，后来去了美帝音讯全无……

你们说，这些人能复制和培养么？

高手不可培养，只能独自成长。如果你的团队里有这样的高手，好好珍惜他们并给予他们足够的空间，也许有一天会有数百倍的回报！

老兵不死，只能自我提升

【发布日期　2013年7月31日】

按照现代人的生存环境，我觉得70岁可以算人生的有效年龄，所以说35岁人生过半并不为过。对于喜欢思考的人来说，人生过半还能够听进人言的，我觉得有圣贤之心，当然能否成就圣贤之事要另说。常年读书、思考，并每日三省其身的人，30多岁基本都形成了自己的完整三观，至于三观正与不正，就仁者见仁了。这样的人你试图以旁观者清的理由告诉他一些人一些事，提些建议或给些意见，得到的反馈基本都是面带笑容频频点头和若有所思，心里不骂娘、手里不攥紧诺记手机已经算是对你客气了。

我和身边的朋友、同事探讨过这个问题，大家都有同感，一个问题只有自己想清楚了，悟了，才会去回顾之前别人的那些碎碎念，哪些是建议哪些是垃圾，哪些是牛人哪些是吊丝，然后给别人分分类排排序，以为自己下次会更多采纳那些优质建议，其实下次还是一个德行。伟大如乔布斯者，经常对苹果的那些牛人提出的建议、意见、创意和想法抛出硬邦邦的两个字，狗屎，掷地有声后飘然离去，两周后一准喜滋滋的把大家召集在一起说，哥有个好主意balabala……这时就会有个倒霉蛋在后面骂娘“这特么不是我两周前的点子么？”说起来都是眼泪……

注：看了这篇文章不要试图和你的领导去探讨意见和建议的事，无论是工作中的还是家里的，他/她们的处理原则是不变的：有意见可以提，保留就是了。不听劝非要去争取自由的，生死由命富贵在天，MacTalk不负担任何法律责任！

其实更稳妥的做法是让他/她们订阅MacTalk……

为毛会这样，给大家讲个故事先：

话说唐僧师徒四人取经成功位列仙班，第一天去上班时进大雄宝殿，被看门的拦住了。大家一看，原来是当年悟空大闹天庭时揍过的一个罗汉，罗汉说：

“虽然你们都成仙了，但是还得回答一个问题才能进去。”然后问唐僧，“你信佛吗？”

唐僧心里说，尼玛信不信都得说信啊，于是回答，“信”，然后就进去了，猪八戒和沙和尚也是这问题，分分钟都进去了。猴子心想，这特么太简单了，正准备整整虎皮战衣抬腿进殿，罗汉问道：

“大圣，你知道世界上有多少只猴子吗？”

“你大爷……”

这个故事当然是我编的，中心思想就是，我们每个人都是那个罗汉，在判断一个人或一件事或一句话的时候，已经根据以往的经验教训进行了预判，为不同的人和事设立了不同的准则，我们只相信那些愿意相信的东西。这玩意有个术语叫做“确认偏误”，英文叫Confirmation Bias，一旦有了这种偏见，我们就很难改变主意和接受他人意见了。

比如那些愿意相信气功大师的人，如果你说气功大师都是骗子，他就会问你“猴子”的问题，如果你说气功大师是国学国粹的传承者，那么他就会问你“佛”的问题。如果你说气功大师能隔几条街戳死谁，他就会两眼放光说，我擦不是真的吧……

现实如此，网络上更是乌烟瘴气，网络骂战几乎分不出胜负，大部分人都在相信自己原来相信的东西，只有那些一直积极思考并且愿意接受新思想的人才有可能改变，这部分人的比例是极少的，如果你是，那就是大幸！

写到这我对古代那些当谋士的人充满绝望的缅怀，想碰到一个上马可杀敌下马可治国还能纳谏如流的明主，基本上和现代中彩票的几率差不多，悲乎！

想清楚这些之后，我目前的行为准则是这样的：

1.如果别人需要我的意见，知无不言，但不要去追寻结果，不要以为你的建议是金科玉律不听不行，更不要事后说，“我早就说过……”。

2.如果我需要别人的意见和建议，那就把自己放空，暂时扔掉偏见和预判，认真听好好学，争取融会贯通，提升自我，如果确实无法接受，那也不必抱怨，最后要记的说“谢谢”。

老兵不死，只能自我提升，人生能做到大通达，小拧巴，就行了……

留不住的人才

【发布日期　2013年2月19日】

工作以来虽然一直从事技术研发工作，但很早就涉足所谓的“管理”工作，掐指算算，带团队有十年了，人多人少，人聚人散，有时想想蛮伤感的，很多非常好的朋友兄弟，因为工作分开后慢慢的疏远再不相见。这可能就是古人两千年前所说的“相濡以沫，不如相忘于江湖”。春节后第一天工作，就来谈谈这些留不住的人才。

几乎每家公司无论大小都在标榜——以人为本，但真正做到的凤毛麟角，甚至用“几乎没有”来表述也不为过。在这样一个信息时代，人才几乎就是每家公司的所有财产，所以各大公司都希望能够留住宝贵的人才，但恰恰是信息的流动也导致了人才的过度流动，这就像各大球队转会一样，每到年中年末，不停地换人、裁人、进人，现在IT行业的人员流动比转会更加剧烈。

我带团队最好的成绩是在2008-2010年，两年多的时间里团队流失率为零，当时沾沾自喜，觉得自己具备带人的天赋，实际的情况是，虽然我采取了很多人性化的做法，包括定期主动为优秀员工加薪的措施，但主要的原因是因为互联网和移动市场还没有爆发性增长，从2011年开始，电商、移动、搜索、云计算、大数据，乱花渐欲迷人眼，热钱已经迷了眼，整个行业的薪酬水平已经完全无法预测了，IT人员开始疯狂的跳槽，所有的人才都是留不住的人才。

关于跳槽和薪资，这个扯起来就扯不清了，坚守还是离职，长远利益还是短期利益，对于那些背负着房贷车贷和没有房贷车贷的80后90后，任何选择都不应该受到苛责。所以我们就知道，人才是留不住的，除非他们想自己留下，不断的引入新生血液，培养他们，即使他们两年后会义无反顾的离开……

最后用伟大的披头士乐队的最后一张专辑的名称来结束这段碎碎念：Let it Be！

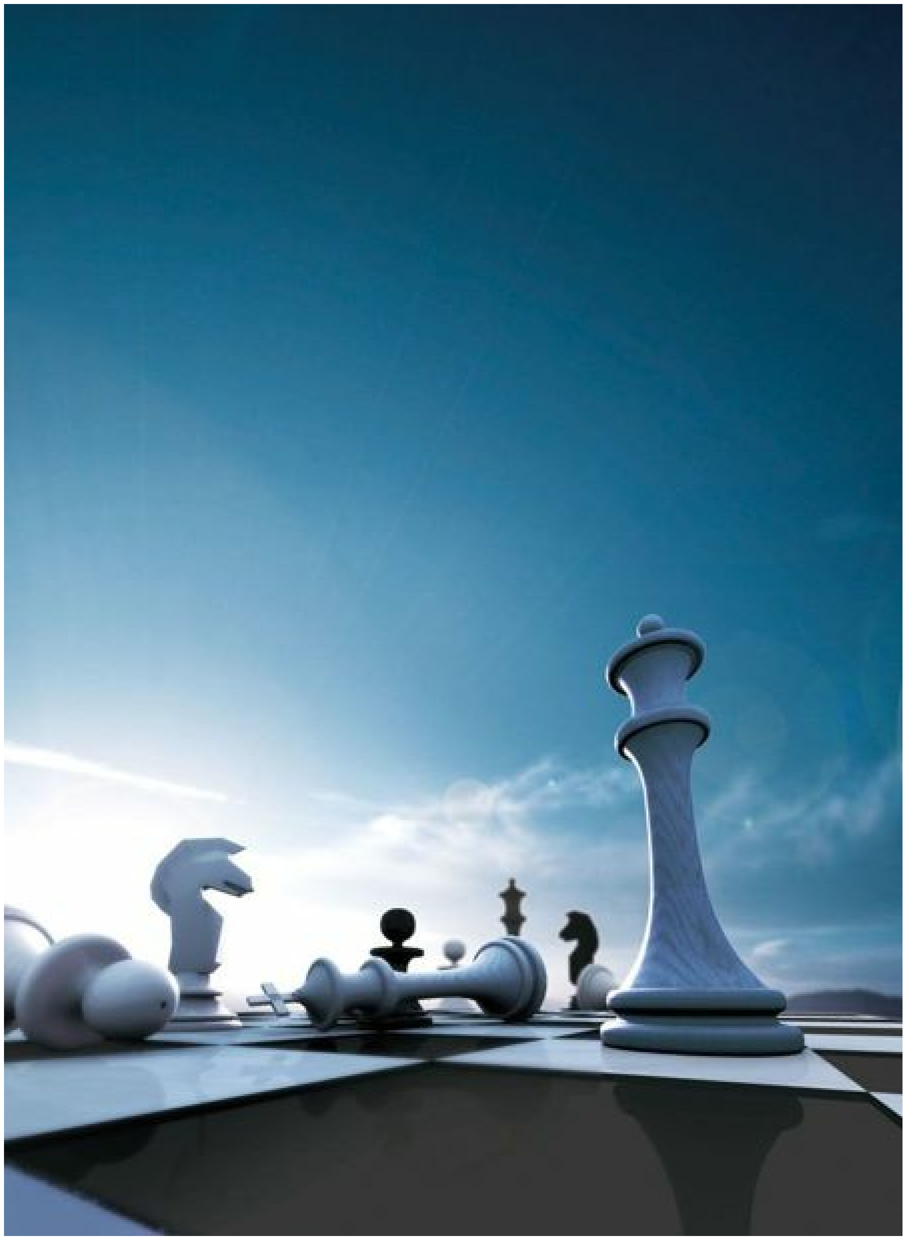
没文化有人文

【发布日期　2013年6月5日】

看到一篇文章，叫做“不穿裤子驱动开发”，看题目以为出现了牛叉闪闪的新型开发方法，细读之下，发现是一个没见过世面的外国程序员的碎碎念，大意就是他们可以穿短裤上班编程，然后有个兄弟就发明了个词叫做PDD(Pants-less Driven Development)，我觉得人家是为了讽刺各种DD（比如FDD-特征驱动开发，DDD-领域驱动开发，TDD-测试驱动开发balabala……），结果文章作者把这事扯到企业文化上了，比如快乐、幸福、自由等等。

好吧，这篇文章就拿企业文化开刀。

我记得最早参加工作的时候没听说企业文化这回事，那时候墙上都写着团结、紧张、严肃、活泼这样自相矛盾的词汇，所以大家都视而不见。后来不知道怎么回事，估计是美帝文化入侵，国内大大小小的公司都开始有文化了，大家使用了各种正确的废话来作为公司的文化和口号，比如创新、严谨、务实；比如诚信立足、创新致远；比如为用户创造价值；比如与客户共赢balabala……，还有更奇葩的，比如态度决定一切，细节决定成败；比如态度决定行为，行为培养性格，性格决定命运，到最后你都不知道是谁他喵的决定了谁，怎么看怎么像死循环。



还有不少企业把公司的愿景作为企业文化，比如基业长青、百年老店等等，但是如果没有把公司的利益和员工利益整合在一起，那这个文化就是老板的文化，企业投资方的文化，和员工有毛关系？

还有企业搞文化是为了给公司员工洗脑，把员工招进来以后先拉到一个鸟不拉屎的地方进行集中培训和宣讲，告诉员工，进了这家门就是这家人，你必须得和我们长的一样，否则就要一刀两断大义灭亲！

还有企业搞文化是为了推行胡萝卜+大棒+大饼的政策，通过企业文化画大饼，然后辅以考核、绩效、奖金，一套组合拳下来，估计新老员工就晕了。

还有更奇葩的企业，在不同的阶段会不断变换自己的企业文化，就像人说“历史是个任人打扮的小姑娘”，企业文化也成了小姑娘，不同阶段穿不同的衣服，您的情商如果没有达到某个高度，是很难理解这种转变的。

其实我觉得企业文化这事挺好，做好了可以决定企业的基因，比如谈到创新、Think different，我们就会想到苹果，提到开放、Don't be evil，我们就会想到谷歌，谈到工程师文化，我们就会想到Facebook，但这玩意有时也会带来一些困惑，比如苹果没发布新产品，很多就是说你创新已死，其是大家掐指一算，从iPod、iPhone、iPad之间都间隔了好几年。Google和别人打专利官司，关掉大家喜欢的服务，也会有人说，你丫不是不做恶吗，我呸~~~~~~。Facebook上市后市值大跌，结果有人喷工程师文化完蛋了。

我要说，这些外界的声音都不重要，重要的是企业是否真正认可自己的企业文化，如果是，继续保持就好了，苹果依然会创新和不同，谷歌保持开放，Facebook工程师主导，他们甚至不需要去解释什么。

反观国内，很多公司把企业文化作为一种“秀”或“工具”，这就比较扯淡了。我觉得搞好企业文化，就两点：

1.利益，把公司利益和员工联系在一起，好员工钱得给足。

2.人文，少搞或不搞办公室政治，让员工自由一点、开放一点、平等一点，你会获得回报。

对于员工本身来说，不管企业是什么文化，如果你热爱自己做的事情，并愿意付出劳动获取酬劳，那就去做好了。

当你在夜里独自思考，反复打磨你设计和创造的东西，在清晨的微光中看到自己的作品，我想告诉你，你为之付出的每一分钟都是值得的。

缅怀那些沉没的项目

【发布日期　2013年7月4日】

上一期谈了沉没成本的问题，很多读者表示意犹未尽，非要问我哪些项目沉没了，这可谓哪壶不开提哪壶，看热闹不怕事大，事不关己高高挂起，既如此，那我就缅怀一下我经历过的那些失败的项目和沉没的成本，让你们的周末更加愉悦一些。



互联网的泡沫

1999年底，我离开北京的郊区顺义，离开堆满散热器的车间，来到北京，看着北京四环上一路的滚滚红尘，我心想，这才是我要的生活！2000年，初入IT江湖，伴随着第一波互联网浪潮的冲击，我第一个真正参与的项目是洪恩在线。洪恩软件当时就在五道口的东升乡楼里，对面就是清华大学东门，公司学习和学术氛围很浓。那时使用的编程语言是Perl、Shell、HTML和JavaScript。那段时光美好的令人发指，全公司80％的员工投入在网站建设上，其中大部分是程序员，大家都青春年少一腔热血，持剑四顾眉宇苍茫，每个人都像贪婪的海绵吸水一样吸收知识并拼命编程。

当时我们的日程表是这样的：上午11点左右，一堆睡眼惺忪的程序

员踢踢踏踏走进办公室，打开自己的个人电脑，勤奋的人会噼噼啪啪的编写一会程序，慵懒的就会浏览一下网站的流量和论坛评论，等到12点去食堂吃饭，吃完饭休息一下，下午正式开始一天的工作，路遥当时写《平凡的世界》时号称“早晨从中午开始”，我们同样如此。每天的下午和晚上是大家效率最高的时候，那时候只听到噼噼啪啪的键盘敲击声和嗡嗡的讨论声，这种情况一直持续到晚上九点左右，有人开始离开，我们一般会继续编程到凌晨，每晚12点左右要更新程序和网站内容，做完这件工作之后，大家就会放松一些，开始看书和学习一些前沿的技术，到凌晨两三点的时候，就是我们每天的happy time，几个人开始联网打对战游戏，比如三角洲、雷神之锤等，慢慢天光发亮，我们就知道睡眠的时间来临了，各自蹿回公司的宿舍睡觉，等待下一天的来临。

那段时光是如此美好，以至于我开始怀疑它的真实性，无论如何，我在那个阶段得到了肆意的成长，身边都是清华北大的牛人，只要你想学就会有无数的新鲜玩意等着你，每天我们都被新的技术和潮流冲击着，并幻想这种生活能长长久久的持续下去……

但是，幸福的时光总是短暂的，接下来便是无穷无尽的痛苦！很快互联网的泡沫破灭，无论投入了多少，公司必须要考虑自己的现金流了，大批的人撤走重新去开发教育产品，但是网站开发积累了大批的程序员如何处理？总不能把我们收拾一下埋在清华园里，虽然可以埋葬回忆，但埋人显然是不对滴，为了解决这个问题，公司决定新启动一个项目，叫做数字校园，准备使用最新的技术Java实现，新的一段历程又开始了。

洪恩在线经过大跃进式的轰轰烈烈之后悄然沉寂，究其原因就是随风而动，但是那个年代因此交学费的，可不止洪恩软件一家，所以这个沉没的成本，可以归结为历史原因。

数字校园

“数字校园”当时的定位是基于BS架构的校园信息化平台，大部分功能使用浏览器操作，少数类似排课引擎要安装客户端。主要技术架构J2EE，支持Windows、Linux、Unix，数据库采用PostgreSQL，Http Server是IIS+Tomcat或Apache+Tomcat，主要语言是java和jsp，其他用到的语言包括汇编、C、VC、Shell等，当然前端的HTML和JS是少不了的。

为毛会扯到汇编上呢？这个问题我也郁闷，汇编是我当时的老板写的，我之前提到过这位“三无人员——无论操作系统、无论编程语言、

无论数据库”，他的风格是能把简单的事情搞的巨复杂，也能把复杂的技术搞的巨简单，当然最终的结果是把问题搞定，把菜鸟搞晕。当时那些汇编程序应该是用来做产品License验证的，但现在想来我觉得他的精力更应该放在卖License上而不是保护License上。

为了在浏览器上模拟客户端程序的效果，我们使用xml数据岛的技术实现浏览器异步加载，后来类似的技术有个更时髦的名称叫AJAX。为了实现智能排课和分班，我们走访了很多学校，编写排课和分班算法，分别实现了在线版和离线版……

经过以上结案陈词，我们就知道，在这个案子里，无论是技术水准还是业务理解，我们在同领域都处于领先的地步，我们的团队对成功是如此笃定，就像一个优秀射手在连续错失多次良机之后，面对空门我们相信这个一定会进……我们猜对了开头，却想错了结局，球再次滑门而过，射手爬起来泪流满面，md原来是国足射手！

总之，数字校园并没有给我们带来好运，虽然我们的产品不错，但是理念超前，曲高和寡，洪恩本身缺乏企业级产品的运营和销售经验，学校的信息化程度也没那么迫切，在坚持了两年的研发和销售之后，最终这个产品也以缩减编制告终，我在这个阶段也开始成长为一个技术管理者，帮助产品负责人协调各种技术牛人的工作，当然我编写的代码并不比别人少。但是项目的沉没给我带来的挫败感远远大于成长的喜悦，对此我无能为力。一切似乎都没有落在“点”上，沉没没有注定却依约而来，让人唏嘘感慨！

由此我也想明白了一个道理，通晓天下武功而百无一用，其实和不出家门的书生也没太大区别；学尽世上技术但做不出成功的产品，一样是毫无意义。所谓知行合一，实在缺一不可！

当培训遭遇非典

时间来到2003年，公司老板目光炯炯，准备开辟培训市场，他告诉我们失败不可怕，目光坚定勇往直前一定能成功。我当时的老板具备一定的现实扭曲力场，我们透过婆娑的泪眼看到他坚毅的面庞，再一次相信了真心与梦想，我们开始规划培训帝国的荣光……

洪恩教育当时的英语培训力量还是很强的，为了做出差异化的培训，除了准备师资力量和多样化的课程之外，我们还设计了一套英语培训系统，系统不仅可以记录学员的信息、学习进度、学习课程，还可以通过视频进行对话和练习，讲师会对学员的文字作业和视频作业进行点评，形成了一套教学、视频反馈、文字反馈、点评、成长、教学的闭环

培训体系，这套系统的开发任务主要由我和另一位程序员完成，那哥们是清华的硕士，人称阿黄，一手VC写的十分俊朗，经过严肃的讨论后确认，用C#写培训系统，VC实现视频引擎。当时我们的效率还是很高的，系统很快上线，场地、学员、设备一应俱全，一切都在往成功的方向挺进，然后，就非典了，然后就没有然后了……

我们的状态从打鸡血直接转换成泄了气的皮球，乾坤大挪移也不过如此。我躺在宿舍望着天花板痛苦的思考着中国IT的未来，心想，这行业到底是不是人干的？看看吧，Mac君年轻的时候就开始思考这些木有答案的问题了，各位初入或将入IT江湖的师弟师妹师侄，我只能帮你们到这里了……

我经过长时间的思考之后，做了一个看似“错误”的决定（兄弟们，所谓长考出坏招就是这样滴！），洪恩依然在不断寻求突破和机会，但我已经疲惫不堪了，最后放弃了做管理和做游戏的尝试，离开洪恩去做其他事情。结果原来那票兄弟2004年成立完美时空，度过了一段极其艰苦的岁月之后，成功开发出了诛仙、七侠镇等经典游戏，并于2007年在纳斯达克上市。呜呼，这也就是坚持的力量吧，美好的结局总是送给坚持到最后的人们！

之后我又经历过一些失败的项目和沉没的成本，不再一一细数，总之，技术不行、时机不对、策略失误、公司基因等等，每一个因素都可能会导致项目的沉没，而坚持，有时降低成功率，有时提升成功率，时也命也，就是如此！

最后用一首送别的词来送别这些沉没的项目：

登山临水送将归，悲莫悲兮生别离，不用登临怨落晖。昔人非，惟有年年秋雁飞。

明天的科技

【发布日期　2013年7月22日】

每次写完一个规划中的长篇文章都会比较累，今天微信宕机大半天，我高兴坏了，心想一直坏下去今儿就不用写了，没想到天不随人愿，临时工愣是把电缆接好了，好吧，今天扯点轻松的话题。



在人类漫长的历史长河中，科技的发展其实一直是缓慢的，从穴居山洞、茹毛饮血，到拔剑四顾、踏马江湖，再到汽车飞机、一日万里，整个过程历经上百万年，科技就像一头年迈的耕牛在时光的地垄间缓缓前行。直到20世纪中叶，计算机被整出来了，随后互联网诞生，从此人类的科技开始快马轻裘，进入了发展的快车道。昨天还是电灯电话楼上楼下，今天已经是智能手机和平板电脑的天下，明天的科技属于谁呢？Google的Glass和无人驾驶汽车，还是Apple的可穿戴智能设备？亦或是马斯克的超音速管道和火箭飞机……

有诗吟道：北方有佳人，绝世而独立。一顾倾人城，再顾倾人国。今天我们还能找到倾国倾城的科技么？今日佳人，明日黄花，2010年被推崇备至的iPad和iPhone已经被批评没有创新了，更早的微软和IBM已经被说成是没落英雄了，时至今日，说英雄谁是英雄？

上次和Google的朋友聊天的时候他们让我评价一下Google和Apple，我说，这两个公司开发出了我每天使用最多的产品和服务，比如Apple的iPhone、iPad、Mac和App Store，Google的Search、Gmail、Analytics、Map等，所以它们是我最喜欢的两个公司。

Google呢，精深而充满未来感，他们有世界上最庞大的数据、平台和搜索能力，但却并不迷恋于此，而是把领域拓展到未来，开发出了Google Class和无人驾驶汽车，虽然这两个产品的盈利还不可预期，但是很多人看到了Google的眼界和探索精神，这也是其股票一直稳步增长的原因。

关于Google眼镜我要插一句，为毛老外戴上总是帅帅的，中国人民一戴总是土气四溢傻气逼人？我觉得要么是这货专门为老外脸型设计了，要么是国内那些眼镜男都不够洋气。哎，有时我被自己崇洋媚外的心理活动搞的什么都吃不下，最后只好去吃海底捞。

Apple则是不创新就去死的公司，很多人指责苹果这几年没有创新，但是不要忘了，创新是需要时间的，在你刷微博吐口水扔臭鸡蛋的时候，真正的创新者都在踏踏实实的用锤子钉钉子，用电烙铁焊板子。如果我们从2000年开始回顾的话，Apple的每个创新大概都要经历3-5年的时间，2001年的iPod，到2006年的Macbook，2007年的iPhone，2010年的iPad，现在是2013年，我想是到了创新可期的时候了（iWatch？）。

一个产品从无到有是困难的，从有到精是艰难的，而当你站上一个巅峰之后，哪怕是做最微小的改进和提升，都需要花费大量的人力物力，同时还要承受失败的风险。我们都知道，可能很多人、很多公司穷尽一生都无法达到卓越的境地。所以对Apple来说，我们只是需要一点耐心，在把我们自己的事情做好的同时，再等等。

今天之所以想写这个话题，是因为这几天正在玩Jawbone Up手环。智能穿戴设备似乎已经成为未来的主流趋势，但是Google Class在国内似乎还是个高端玩具，尤其是我们这种读书读成近视眼“高端人士”，看见别人带Google眼镜耍酷心中就有几万头泥马翻腾奔涌。iWatch呢，除了一堆效果图毛都没有，所以玩玩健康手环是个不错的贴近未来的选择。

我个人除了对Apple的产品抵抗力稍弱之外，在其他方面基本上土的掉渣渣，自然不知道有这种手环，这是上次参加阿里技术嘉年华获赠的，回来就开始折腾，用了一周多，感觉不错，手环和iPhone或Andorid平台上的App配合，可以记录你每天运动了多少步，睡眠质量，睡眠时间，还有智能睡眠闹钟，空闲提醒等功能。长时间使用还可以记录你的

生命线和趋势。

当然有个功能比较鸡肋就是记录饮食信息和热量，全特么是英文，您要是吃了个土豆炖牛肉，根本就不知道填啥，所以，目前基本没用。

用这玩意的另一个好处就是，如果需要徒步去做点什么，以前的反映是“我了个擦，怎么这么远”。现在的反映是，“好，又可以增加几千步了”。

扯了这么多，明天的科技到底是什么样子呢？没人知道，未来的事情谁说的准呢，不过我觉得，软件与硬件的整合，硬件与人的整合，人与自然的整合，可能是未来科技的发展之路吧……

你有多少时间？

【发布日期　2013年5月22日】

经常有读者问我每天花多少时间在写作上，你怎么会有那么多时间呢？

其实我一直在压缩写微信平台的时间，包括取消了每天更新的承诺（于我而言，时间确实是不够的），但是固定的时间消耗必不可少，现在大概短则半小时，长则两小时，这里面还包含了读者互动的时间，时间成本还是蛮高的。怎么会有那么多时间？其实没那么多时间，大家时间都是一样的，只不过在做不同的事情罢了。说到这里，我就要给大家介绍一个节省时间的绝招，那就是远~离~电~视！



小时候有个阶段我最爱的两件事就是看武侠小说和武侠电视剧，金庸古龙梁羽生的书基本读了个遍，常常就着夜色和手电筒看书，碰到武侠剧一集都不想落下，估计眼睛近视就源于此。看到痴迷处仰望星空，

我就琢磨，等老了就弄两筐武侠小说和录像带看，想躺着看就躺着看，想坐着看就坐着看，想看多久就看多久，那是何等豆浆和冰糖的生活！

那时候的我还不知道人在各个阶段的变化是如此难以预料，二十岁的时候很难想象自己三十岁时候的思想，何谈老去？梦想，能力，规则和习惯等，都会随着时间流逝发生改变。很快武侠小说就淡出视野，但电视还是充满吸引力，刚毕业时租了房子，首先给自己买了台电视，工作回来后，经常躺在床上看着电视入睡，那时的网络还没有如此发达。慢慢我开始意识到电视这玩意简直就是时间杀手，光换频道你就能换一个小时，别提那些垃圾广告了，后来慢慢电视也淡了，30岁之后，基本戒掉了这个娱乐活动。现在我家的电视有几百个台，仅用来看看体育直播和重大新闻，偶尔看看电影频道。由于我们不看电视，孩子对电视也没瘾，可以用更多的时间做其他好玩的事情。

据不完全统计，现代儿童在电视机面前每天要耗上2-3个小时，这几乎占到了孩子们课余时间的一半还多。各种垃圾广告和言论轮番对人们进行轰炸，画面的不停切换和非互动性会让人们失去注意力，当然更大的损失是自己的时间。就我个人的了解，现在的电视几乎没有任何营养，90％的娱乐内容、脑残电视剧和无比和谐的“你幸福吗”，我觉得不看也罢，人类通过书籍留下了那么多美好的东西，即使你不想提升自己的专业能力，看看各个时代的书，你也会知道宇宙洪荒，桑田沧海，人类的各种美好和苦难，看什么电视呢？

确实想看的，看看美剧，顺便还练练听力。

当然，网络也是个时间大杀器，同样需要控制，但网络至少比电视好，至少具备互动性，至少你是在主动获取内容。

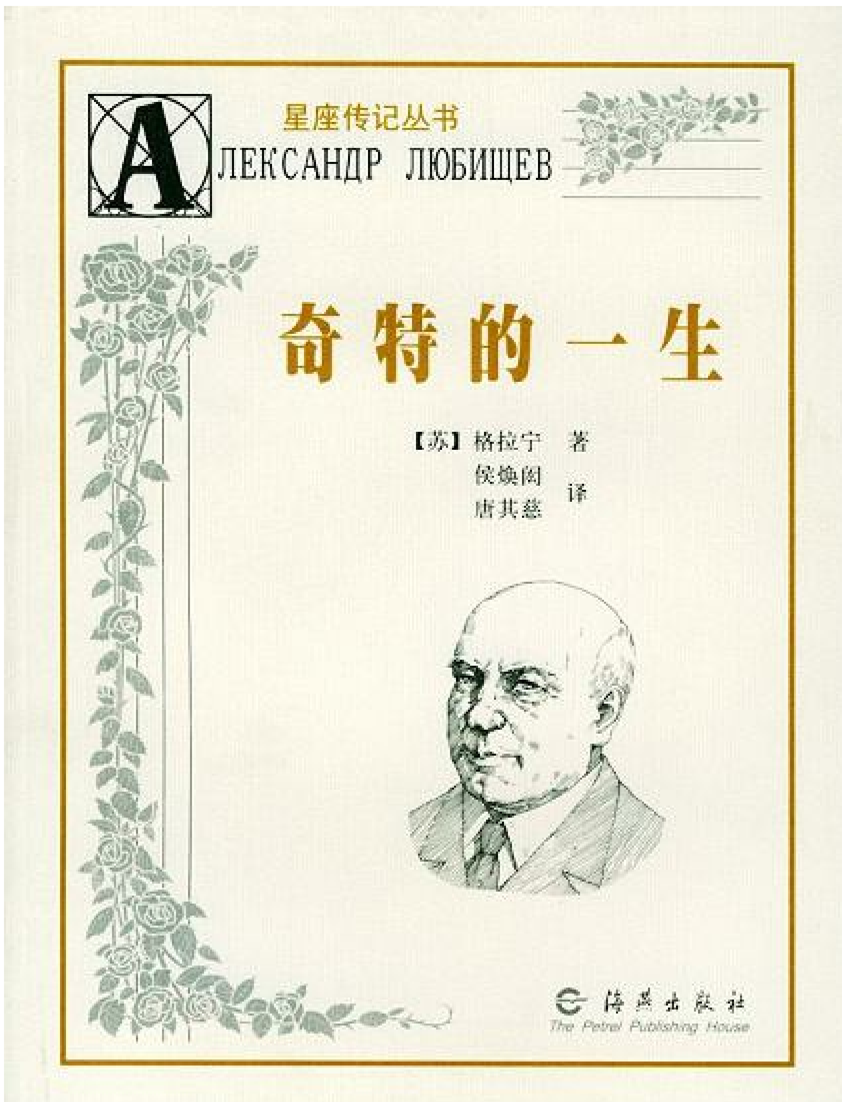
如果你现在正躺在沙发上看电视，那么最好关掉电视，好好去读一本书。

奇特的一生

【发布日期　2013年3月8日】

今天为大家介绍一位俄罗斯的科学家，但在之前首先谈谈我对全才和专才的理解。

很多人的看法是二选一，要么全面发展，样样精通样样稀松；要么成为某个专业领域的顶尖专家。作为一个70后程序员，Nac君万不赞同！



就拿IT技术来说，工作十年以上的，想不成为通才是很困难的，除

非你确实是不思进取，而且十几年不跳槽。很难想象一个工作近10年的技术人员说，我只会Java！回顾一下，从2000到2013，技术领域的革新速度令人瞠目结舌，说日新月异不为过。各种操作系统、编程语言、数据库（Sql的、NoSql的）、各种框架、平台、业务系统，接踵而来，数不胜数，你运气得多好才能做到十年只用一种操作系统、编程语言和数据库啊？

比较好的方式是全面熟悉你接触或主动学习的内容（非浅尝辄止），同时在其中根据兴趣和方向打造自己的专项特长。

写到这估计就有同学要发言了，那位已经涨红了脸的童靴你说一下吧，“Mac君，全面学习打造专长，说者容易做者难，不吃不喝光学东西，你当我神仙啊？”好的这位神仙，你先坐下，把手里的臭鸡蛋放到桌子里面，下面我将用一个完美的公式证明这是可行的。

是啊，人怎么会有那么多时间学习那么多东西呢？其实这个不可能的设定，是在保证你有足够时间看电视、看美剧、刷微博、上网闲逛的基础之上的。只要把上述这些事情消费的时间减少一半，拿来持续学习，你就会发现学习效果是惊人的。

有一本书叫《奇特的一生》，介绍了一位叫柳比歇夫的科学家，这个兄弟生前发表了七十来部学术著作。其中有分散分析、生物分类学、昆虫学方面的经典著作；各种各样的论文和专著，他一共写了五百多印张。等于一万二千五百张打字稿。涉及内容包括：著作，探讨地蚤的分类、科学史、农业、遗传学、植物保护、哲学、昆虫学、动物学、进化论、无神论。此外，他还写过回忆录，追忆许多科学家……

柳比歇夫为什么会有如此多的时间干这么多事情？难道他不睡觉不娱乐每天工作18小时么？非也，很多宣称自己每天工作15小时的人，真正有效的工作时间可能不到一半。柳比歇夫把时间分为纯时间和毛时间，纯时间是要把工作中的任何间歇都要除去的，他这么描述：

> “常常有人说，他们一天工作十四五个小时。这样的人可能是有的。可是拿纯时间来说，我一天干不了那么多。我做学术工作的时间，最高纪录是十一小时三十分。一般我能有七八个小时的纯工作时间，我就心满意足了。我最高纪录的一个月是一九三七年七月，我一个月工作了三百一十六小时，每日平均纯工作时间是七小时。如果把纯时间折算成时间，应该增加百分之二十五到三十；我逐渐改进我的统计，最后形成了我现在使用的方法……”
>
> “当然，每个人每天都要睡觉，都要吃饭。换句话说，每个人都有一定的时间用在标准活动上。工作经验表明，约有十二——十三小时毛时间可以用于非标准活动，诸如上班办公、学术工作、社会工作、娱乐，等等。”

这个人对时间有精准的把控，他会把自己每天的工作和用时记录成册，时间就像经过严格控制的沙漏一样，在他身边缓缓流过，他仿佛能

感知时间的流失。这样的自控能力、学习和工作效率让人叹为观止，常被我奉为天人！每当偷懒时总是对自己说，孙子，你看看人家柳比歇夫子是怎么干的！当然，咱不是天才，有时候懒还是要偷的……

所以，综上所述，无论你现在处于哪个境地，是手握优质资源还是即将被裁掉，其实都是自己之前的选择造成的。我们不能改变出生的国家、年代、家庭背景、运气，我们唯一能做的就是充分利用时间，按照自己的兴趣把自己训练成通才之上的专才，这个道理很多人懂的晚，做的晚，但是，就像写MacTalk一样，写的慢不要紧，重要的是持续的写！

证明完毕！

是旅行还是长跑？

【发布日期　2013年6月26日】

你苦战通宵游戏时，布里斯班的灯鱼已划过珊瑚丛；
你赶场招聘会时，蒙巴萨的小蟹刚溜出渔夫的掌心；
你写程序代码时，布拉格的电车正摇着铃晃过金色夕阳；
你挤进汹涌的食堂时，哥本哈根的街头画家完成了第99幅立体画。
有一些穿高跟鞋走不到的路，有一些喷着香水闻不到的空气，有一些在楼宇里永远遇不到的人。

大伙看到这段文字，一定以为我要炖一锅热腾腾的心灵鸡汤了，然而事实的真相是，Mac君不会做饭！

这段文字在网络上流传甚广被誉为佳句，其实还有很多类似的文字，比如“放下一段过往，期待一场盛大而华丽的远行”，比如“生活在别处，远离城市的丛林”，比如“背起行囊去远行”等等，总之就是一边戳着你的脑门一边对你说“你看看你现在过得是什么生活，你对得起谁，你为什么不把这些破烂扔掉去面朝大海春暖花开？”



这时候你是不是就忧伤啦，躺在租来的小屋里，手边是一个二手的电脑和已经翻的卷边的编程指南，想想蒙巴萨的小蟹和布拉格的电车，你对自己说，这特么是人过的日子么？于是就开始自怨自艾，懊悔不迭，为什么当年特么的不让自选爹？但是想归想，除了极少数的二愣子，大部分看完这些话就是忧伤一下，然后洗洗睡了，因为明天还要去找工作或去写代码。

其实每个人在某个城市呆久了都会疲惫，每份工作干多了都会厌倦，为了疲惫和厌倦的改变，那不叫改变，那叫逃避。面对无数的选择和变化，你沉溺其中，但是依然无法获得理想中的生活。

经过这样一番理性加感性的描述之后，你就知道这种文字除了让人感受令人愉悦的忧伤和丧失斗志之外毛用没有，基本上可以定性为精神鸦片！只要你没有一个强悍的爹，就要先忘掉那些小鱼小蟹还有珊瑚丛，那些暂时还不属于你，踏踏实实的练习和提升才是正道，为自己规划一个十年的长跑，可以时不时停下来休息一下，低下头汗水就落入尘

埃，抬起头就看看夕阳西下，你抹去疲惫，然后继续前行。十年看似很长，但实在很短，十年以后你回头看看，你究竟是成了人中龙凤，还是小鱼小虾，是你认知了世界，还是世界抛弃了你。

如果你真的烦透了当下，想出去散心旅行的，看看这篇《旅行，写作，编程》（http://www.aqee.net/traveling-writing-programming/），同样是在全球各处行走，这位兄弟游山玩水的成果是：写了两本关于JavaScript和CoffeeScript的书，参加了一个技术会议，写了大量的开源库，筹划了一个创业公司的框架，旅行结束后背着包去Twitter上班了，那一年他21岁。这叫游历生活。如果你也可以，那就凡事趁早，早去早回，对了，别说我没提醒你把背包里的PC换成Mac！

零零碎碎写了一些东西，是因为最近见了一些悲欢，看了一些生死，我越来越感觉到，生活更像长跑而不是旅行，你不断的前行和奔跑，在长跑中告别青春、告别幻想、告别岁月、然后慢慢开始告别各种人和事……

你不知道什么时候是终点，可能倒下的时候就是终点吧……

西塘古色

【发布日期　2013年4月2日】

昨天发了一个西塘的预告，很多人以为我去旅游了，其实是去嘉善做一个产品和技术交流。去了嘉善不去西塘，那就像来杭不见西湖，去京不登长城一样，虽然时间非常紧张，但西塘之行势在必得，所以我们把中午的时间放在了西塘古城，匆匆一瞥，意味悠长。



西塘是江南六大古镇之一，位于浙江省嘉兴市嘉善县。六大古镇分别是周庄、同里、角直、西塘、乌镇、南浔，个个是清丽婉约的水乡古镇风貌、古朴典雅的吴侬软语风情，诸位看官有机会都可以去走走看看。

曾经多次看到西塘的风景图画，也在碟中谍3中看到过汤姆·克鲁斯迅疾奔跑的场面，但真正到了这个千年古镇，还是让人感觉闻名不如见面。我想象这个水乡小镇应该有一个很正式的入口，各种美景慢慢入境，结果是，前一分钟还走在嘈杂的城市街道，转过一个胡同，西塘古镇突然就呈现在你眼前了，让人感觉非常突兀。

由于是周一，游人并不多，大家缓缓行走在岸边和蜿蜒的烟雨长廊，与小桥下的流水相得益彰。一进入这个场景，整个时间似乎都变得

慢下来了，长廊流水白墙青瓦，整个西塘就像一幅活动的水墨丹青，对于我这样的北方土鳖来说，一路走一路啧啧称奇。各种形态各异的房屋依河而建，无论是白墙青瓦，都错落有形，各有各的精致。

与其他古镇不同的是，80多平方公里的西塘目前依然是一个居住区，带我们去西塘的当地朋友本身就有一个老房子在西塘，他告诉我们夜里的水乡尤其幽静，在这住几晚，就像穿越了时空，远离都市喧嚣和繁杂，人心也会变得宁静。

临走时一位当地居民跟我说，每次见到游客赞叹西塘的清丽古朴都非常不解。因为他们天天居住在这里，已经熟视无睹了。这让我想起原来说过的一句话，旅行就是从一个自己待烦了的地方，跑到一个别人待烦了的地方。此言不虚。

晓说不小

【发布日期　2013年4月17日】

这段时间利用上下班的时间，把高晓松的《晓说》第一季全部听了一遍，晓说虽小，但格局很大，在这里给大家推荐一下，有心人可以去找mp3听一遍，涨见识不费电。



作为一个典型的70后，高晓松这个人我当然是在听了《同桌的年》，《睡在我上铺的兄弟》，《白衣飘飘的年代》等等校园民谣之后才熟知的，当时的感觉是此子才华横溢但恃才傲物，歌里表达的是内向伤感，外在却神采飞扬。后来才知道这位兄弟端得是根正苗红，外公张维是深圳大学的创办者、中国工程院、科学院两院院士；外婆陆士嘉是中国著名的流体力学家、教育家；舅舅张克潜是著名的物理电子学与光电子学科学家；母亲是著名的建筑学家张克群，老爹最不济，也是清华大学教授。基本上是脑门上刻着“高知子弟”四个金光闪闪的大字出生的，小时候对门住着梁思成（梁启超之子）和林徽因，还去过共和国将帅家里玩耍，见过传奇大奖粟裕将军……哎，这见识怎么比啊，怪不得人家能说一年的脱口秀！

高晓松以前的定位是音乐人、导演、制作人、词曲创作者等等，2012年之后，又多了一个名号，脱口秀主持人。2012年3月，由韩寒命名的脱口秀栏目《晓说》开始播出，每周一期说了近50期，其中涉及的内容包括民国的刺客、镖局、青楼，千年科举，美国的医疗、种族、大选、枪、工会，开国将帅授衔、抗日风云、朝鲜战争、军力对比，大师照亮八十年代，好莱坞启示录等等，内容繁复，博古抵今，精彩纷呈，尽管有些观点我不能完全认同，但大部分是高晓松的独立思考，观点新颖，非常值得一阅。

以高晓松的杂文集《如丧》里的一句话结束今天的内容：

> 迄今为止，我把所有喜欢做的事情都做了，除了恋爱和旅行，都已换成了钱，钱不多，够生活。感谢所有的衣食父母，包括我父母。所有人都老了，再没人死于心。我数着日子和钱，等着永逝降临。

遗失的访谈——岁月无声

【发布日期　2013年5月6日】

最近看完了@七印部落发布的乔布斯1995访谈字幕版。这是1995Bob Cringleg在制作《书呆子的胜利》时对乔布斯的一段非常完整的访谈，当时乔布斯还没有回归苹果，他自己的公司NeXT举步维艰，这段访谈在当年只播出了很少一段就被扔到车库里，直到2011年才重见天日。@七印部落花费5个月业余时间，累计听译11700个英文单词，推出了中文字幕版《遗失的访谈》，感谢@七印部落的工作。



看这段访谈的时候，乔布斯还非常年轻，让人恍惚中感到岁月穿梭时光流转。看完之后才知斯人已去，岁月无声！

今天我为大家讲述一些听来有感的段落，并辅以自己的思考。

1、陈规陋习——folklore

很多时候我们初入江湖，不知深浅，不知道什么事能做，怎么做，为什么要这么做，即使怯生生问了，得到的答复往往是，That's just the way it's done（我们向来这么做），于是我们慢慢也变得成熟、圆滑和懒惰，不再去从深层次思考『为什么要这么做？』因为别人也这么做。慢慢的，这些东西就成为了folklore（陈规陋习）。

一个坏的习惯或传统，可能延续十年、百年、千年，直到那个打破陈规陋习的人出现！

提问、思考和努力工作，你就会抛弃这些陈规陋习，找到布满荆棘也满是鲜花的另一条鲜活的路。

2、为什么要编程

乔布斯建议每个人都去学习或了解一门编程语言，因为编程会教会你另一种思考方式。

为什么要编程呢？第一编程可以帮助我们更加轻松的完成工作，第二编程可以帮助我们更好的思考，因为编程本身就是思考的镜子。 思考最大的价值在于你可以用不同的方式去看待世界，看待世界上发生的事情，而不是局限在自己的圈子里自怨自艾。烦的时候想想宇宙洪荒、大江东去，你的烦恼也许就随风而去了。

3、富有的感觉

乔布斯从小被养父母养大，创立苹果公司的时候几乎身无分文，但是他在23岁时身价就达到100万美金，24岁1000万刀，25岁1亿刀。 但这不重要，他从来不是冲着钱去的。富有之后乔布斯依然生活极简，一生如此。

财富应该是一种资源，允许你去做自己想做的事情，让你去实现那些短期内看不到效益的事情。但是钱永远不是最重要的。重要的是公司、人才、产品，以及产品带给客户的价值。

不为钱工作，然后去挣钱。

4、施乐的宝藏

1979年底，乔布斯去施乐公司参观他们的技术成果。施乐给乔布斯展示了三个项目，一个是面向对象编程、一个是计算机网络系统，最后一个才是图形用户，也是最吸引乔布斯的项目。『那是我一生见过最美丽的东西，很快我意识到所有的计算机都该是这样。』

这三项科技成果在未来的几十年内都将焕发出最璀璨的光华，但是非常可惜，这些光辉都不属于施乐，像施乐这样一座科技金山，为什么最后完全失去光芒，沦为石块，这真是让太人费解，我们看看乔布斯是如何解读的……

5、关于产品

掌控传统公司的人是营销和市场部门，而在官僚垄断科技企业内部（IBM和施乐中枪）也是一样，即使产品部门的人做出了更好的产品，又能怎么样呢？已经垄断市场的公司是不会想到通过产品创新来提高业绩的，想提高业绩他们会觉得还是压榨营销部门来的靠谱，慢慢公司就丧失了打造优秀产品的热情和能力。

在施乐研究院，那些天才的技术人员私底下把管理层叫做墨粉脑袋，就是完全不可交流不可救药的意思。所以，施乐在极有可能成为90年代的微软的情况下，失败了，变成历史。

那么对于产品来说，什么才是最重要的？是不是创意呢？

很多人以为有了伟大的绝妙的创意，就算成功了90％，事实完全不是这样，优秀的创意与产品之间隔着巨大的鸿沟，在实现创意的过程中，会再思考和再加工，想法和创意可能会变的面目全非，你需要调整、让步、修改、完善、梳理、尝试各种组合，形成最终的产品。过程和团队非常重要。

期间乔布斯还举了一个磨石机打磨出美丽石子的隐喻，我就不复述了，有兴趣用Google百度一下。

6、大公司有毒

每个公司都是从小变大，之所以能够变大，就是因为有可取之处，因为某些事情作对了。而公司规模扩大后，大部分公司就会变的因循守旧，他们觉得只要遵守流程就能奇迹般的继续成功。于是管理层开始充满幻觉，推行严格的流程制度。很快员工就会把遵守流程和纪律当作工作本身，而不是去创造伟大的产品。　经验告诉我，优秀的人才是那些一心想着产品的人，而不是关注管理和流程本身。

注意，这里的大不是规模，而是臃肿。

7、关于Macintosh

乔布斯说Lisa失败后，他重组了个小组用来研发Macintosh拯救苹果，但我个人以为，在这一点上，乔布斯有所保留，毕竟当年还比较年轻，事实的真相是……

Macintosh起源于一个微不足道的小项目，项目名称叫做“安妮”，项

目的负责人是杰夫·拉斯金。

杰夫·拉斯金是一位技术天才，也是苹果的第31位员工。拉斯金在自己的博士论文里写道，计算机应该具备图形界面，而不仅仅是纯文本。另外，他还有一个梦想，就是为大众制造价廉物美的电脑。1979年，拉斯金说服了当时苹果公司的管理者迈克·马库拉，成立了一个小规模的项目组用来研发廉价的、同时具备图形界面和命令行的电脑，当时这个项目代号非常悲催的被命名为“安妮”，后来拉斯金觉得太女性化了，于是更换了项目代号，用自己喜欢的一种苹果来命名，叫做麦金托什（McIntosh）。但是为了避免与另一家音响设备制造商麦金托什实验室（McIntosh laborator）重名，拉斯金对这个单词做了些微调，改为大家现在熟知的Macintosh。

后来这个项目几经讨论和变迁，终于被乔布斯盯上了，他取代拉斯金接管了Macintosh项目组，开始了Mac的传奇的征程……

8、卓越和普通

生活中大多数事物，卓越与普通之间的差距不会超过两倍。但是在软件行业，这种差距通常会超过几十倍或上百倍。

不在软件行业的人也许很难理解，大家都在编程，难道卓越的程序员会写出超过普通程序员上百倍的代码么？显然不是这样，但是卓越的程序员可以用简练优雅的代码创造出伟大的产品，卓越的程序员可以花半天的时间解决困扰普通程序员半个月的问题。这些才是真正的差距，卓越程序员创造了百倍普通程序员的利润，但大多数的他们并没有享受百倍的待遇，所以，如果你的团队里有卓越的程序员，那是你的运气！

对此，乔布斯的建议是，构建一个团队只需找到才华横溢和不甘平庸的5个人，让他们形成合作关系，然后他们会找到更多同一档次的人，自动扩展团队，并做出伟大的成就。

9、关于乔布斯

对于乔布斯的暴脾气和经常骂人为狗屎的恶习，乔布斯解释说，在他眼里只有正确的事情，而不是去考虑人情是非和办公室政治。并且他认为牛人有足够的自信，不怕被他称为狗屎。对此我深表疑虑，那得多强大的内心啊：）

10、关于微软

在1995年的这段采访中，乔布斯基本从精神层面秒杀了微软，用一句话形容就是一流经营，三流产品，并且没有品位！他认为微软的产品既没有灵魂也没有魅力，更让人难过的是客户居然没有觉察到这一点。

这一点该怎么理解呢？我来举个例子说说，当年微软的手机操作系统叫做Windows Mobile，这个系统是按照PC操作系统的模式来设计的，有桌面和开始菜单，想进入某个程序，需要用触摸笔点击开始—>程序，在菜单里找到你想使用的程序并打开。如此灭绝人性的使用体验，在开创性的iOS操作系统横空出世之前，使用者的感受是，还可以！

乔布斯认为，人活着是为了追求极致并分享美好的东西给人类，而不是做三流产品并赚钱。这样社会才能进步，让更多的人欣赏到更美好的东西。微软不过是另一个麦当劳，哈哈。

11、关于NeXT

乔布斯离开苹果后开创了另一家公司叫做NeXT，意为下一代计算机。

他认为，计算机产业要创新需要靠软件，但是软件的开发方式在那20年一直没有太大的变化。当时的Macintosh虽然降低了用户的使用难度，但是提高了程序员开发软件的难度。而NeXT公司采用了面向对象编程技术，不仅能够提升软件开发速度，而且质量更好。

从后续十年的发展来看，这个预言更像是给Java做出的预言，NeXT使用的语言Objective-C是在iPhone和App Store发布之后才开始风生水起的。

访谈一年后，苹果收购NeXT，乔布斯创建的NeXT公司的操作系统NeXTSTEP最终变成了苹果的OS X，这真是宿命轮回！

12、展望

乔布斯在1995年对未来的技术发展进行了精准的展望：计算机将不再是独立的，互联网将把它们联结起来；在线电子商务将取代电视购物；Web技术将成为重要的里程碑，其巨大潜力会吸引更多年轻人进入计算机行业。

并且，微软还没有注意到这一点！（连这都预测枭~~~）

结语：采访结束一年后，乔布斯将NeXT出售给苹果，在苹果即将破产之际，乔布斯重新接管了他创办的苹果公司，并开始了美国商业历史上绝无仅有的拯救行动和伟大复兴之路。

随着iMac、iPod、iTunes、iPhone、iPad等创新产品的推出，苹果成为全球最有价值的科技公司。　正如他在采访中所言，他追求极致，分享给人类最好的产品体验，改变了世界！

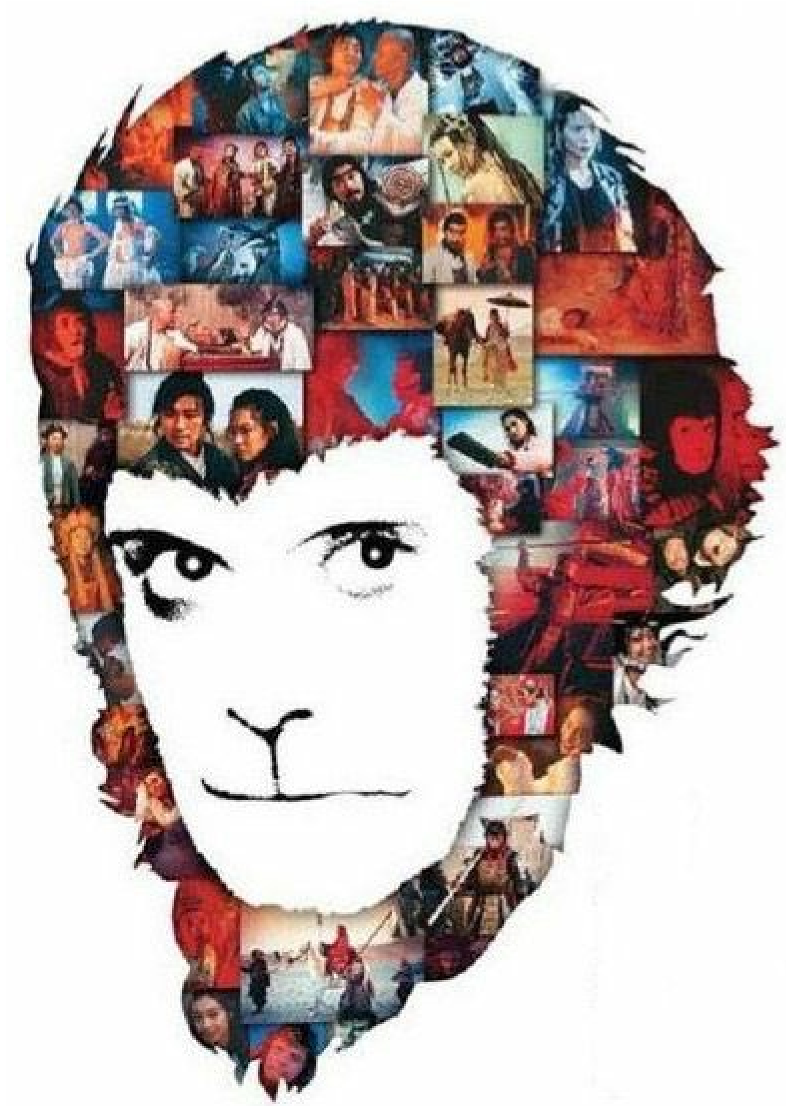
一切都取决于品位！

怎能忘了西游

【发布日期　2013年2月27日】

随着星爷新作《西游·降魔篇》的上映和今何在《西游日记》的出版，似乎再次掀起了新一轮的西游狂想。12年一个轮回，12年前的《大话西游》和《悟空传》给我们这一代70后带来了太多的欢乐和思考，前度西游今又来！

前一阵在字节社（iOS App）重温了今何在的《悟空传-完美纪念版》，除了当年风靡网络世界的网络第一书《悟空传》之外，还收录了花果山、百年孤寂、杨戬传、哪吒传四个新篇，非常好看。另外最近从云中书城下了《西游日记》，花了几个晚上读完了，非常精彩，虽然比不上当年《悟空传》带来的震撼，但依然文笔犀利幽默，在另类西游的基础上对现实社会和人类本身进行了深刻的反思和自嘲，同时少不了今何在文字中一贯蕴含的不朽精神、飞扬的热血和纯粹的理想。



2013年的西游仿佛把我带回2001年岁月，很想为此写点东西，就由此开篇吧。

大话西游

记忆中《大话西游》的《月光宝盒》和《大圣娶亲》分别上映于1994和1995年，当时反响平平票房惨淡，哪个能想到2000年左右在内地大学生中奇迹般走红哩？大话西游一举成为经典，并掀起了声势浩大的大话西游热，同时周星驰被莫名奇妙的推上了后现代解构主义大师的位置，我估计周星驰到现在也没搞清楚这个大师到底是啥意思，其实谁又能明白呢？

当时我毕业没几年，正在洪恩软件工作。洪恩当时的环境就像一所

升级的大学校园，工作环境开放、扁平式管理、提倡工程师文化，管吃管住，写程序还给钱，而且工作地点正对清华大学东门，md，这什么环境啊，于是清华北大的学生纷纷在洪恩实习和工作。那时聚集了大量的优秀技术人才，而且这帮程序员都喜欢看大话西游，我们在公司的电脑上看了一遍又一遍的月光宝盒和大圣娶亲，所有的经典台词都烂熟于心，这些内容贯穿在工作和生活中，新来了程序猿很不适应，于是又去看VCD，我们也乐得再看一遍。

编程时我们也经常会使用大话语言，比如修复了一个bug，会加一行注释："虽然本人生平fix了无数bug，但是这一个我认为是最完美的…"

遇到别人写得方法不好用，会说，"兄弟，你这个API前重后轻左宽右窄，用起来很不舒服，整晚失眠，你就不能改改！""收到！"

俩程序员熬夜写程序，见面语是，"长夜漫漫无心睡眠，我以为只有我睡不着觉，原来兄弟你也睡不着啊！""别扯淡，好好地做你程序员这份很有前途的职业去吧"

现在想来，这些东西为我们当时生活和工作带来了很多乐趣，回忆起来让人倍感温暖。

最初看大话西游的时候，比较喜欢第一部《月光宝盒》，春三十娘和白晶晶打劫脚底板的军事行动相当惊艳，一句"金钱落地人头不保"完爆斧头帮大小头目。那时孙悟空还是至尊宝，在从事斧头帮帮主这份有前途的职业，啰嗦的唐僧、倒霉的二当家、烤焦的菩提老祖和忠贞不二的瞎子，看起来都非常过瘾和轻松，比如那句"娘子，出来看上帝了"让我们笑了一次又一次。第二部大胜娶亲则相对沉重和悲情，但看得多了，反而开始喜欢第二部，你们都知道，紫霞仙子在第二部出场了！

无论是在《大话西游》还是后续谈到的《悟空传》里，紫霞都是一个重要的角色。这是一个寂寞的仙子，充满对爱情的执着和向往，美丽、反叛、痴情，紫霞的扮演者朱茵对这个角色进行了完美的诠释，让人感觉这个角色就是为她而生，能给观众留下这种感受的，除了射雕里的翁美玲，再无第三人。大话西游之后，紫霞仙子彻底成了朱茵的一个符号。如果有人认为我夸张的话，就给你们个证据，去Google或百度的图片搜索紫霞二字，看到满屏幕的紫霞仙子，你们就知道12年前的震撼了，膜拜吧……

《大圣娶亲》的推出同时诞生了那两句流传至今的爱情经典：

第一句是"一万年的表白"。周星驰说了两遍，第一遍是谎言，第二遍说完，带上金箍化身齐天大圣去完成自己的宿命！

这句话我们当年每个人都能背过：

曾经有一份真诚的爱情摆在我的面前，但是我没有珍惜，等到了失去的时候才后悔莫及，尘世间最痛苦的事莫过于此。如果可以给我一个机会再来一次的话，我会跟那个女孩子说我爱她，如果非要把这份爱加上一个期限，我希望是一万年！

据说当年男生都不需要跪搓板的，计算机主板很贵也跪不起，惹女朋友生气后背一段“大话一万年”，就行了。

第二句是“盖世英雄”。紫霞为孙悟空挡了牛魔王的致命一击后重伤，对后悔莫及的齐天大圣说：

我的意中人是个盖世英雄，有一天他会踩着七色的云彩来娶我。我猜中了前头，可是我猜不着这结局……

这句话除了悲情之外，也充满了黑色幽默，当年一位兄弟在集成测试惨遭失败后，痛苦的自责：我以为我是个盖世程序猿，有一天我的程序会奔跑在千万台服务器上。我猜中了前头，可是我猜不着这结局……我们说，呸，你丫从头就错了！

当年的大话西游，为我们这些成天埋头写程序员的年轻人带来了无数的伤感和欢乐，怎能忘了西游！

悟空传

当年除了VCD外，各种解构西游的书籍和文章满天飞，相关的小说层出不穷，然而真正脱颖而出并流传至今的，只有今何在的《悟空传》。就像北乔峰南慕容一样，当年网络文学号称“台湾痞子蔡，内地今何在”。《悟空传》以后，今何在被誉为内地网络文学第一人，可见影响力之大。

我最早知道悟空传并不是从网络上，而是从一个兄弟的网络签名得知，有一天我发现这个木讷的家伙，把水木清华BBS的签名改成了：

我要这天，再遮不住我眼
要这地，再埋不了我心
要这众生，都明白我意
要那诸佛，都烟消云散

料想这厮没有此等文采，一问之下，才知道《悟空传》出版了。

《悟空传》讲述的是悲剧英雄孙悟空以及唐僧等人对命运的抗争，并对高高在上的神仙皇帝进行了无情的嘲讽。其中的情节与《大话西

游》有千丝万缕的联系，很多文字充满了对大话西游的致敬，同时，《悟空传》又有自己新的灵魂，虽然都是悲剧故事和悲剧英雄，《大话西游》是注重痴心与离别，《悟空传》是强调梦想与追寻。

今何在从现代人的角度重新解读《西游记》中与孙悟空的相关情节，通篇文章强化了思考和反叛，弱化了情节，他的文字在那个年代闪烁着叛逆的光芒，其跳跃性和质感吸引了大量的年轻读者，读起来既痛又快。同样，悟空传也催生了大量的经典名句：

> 当五百年的光阴只是一个骗局，虚无时间中的人物又为什么而苦，为什么而喜呢？
> 等到那一刹那，黑暗的天空突然被一道巨大的闪电划开。孙悟空一跃而起，将金箍棒直指向苍穹，“来吧”！那一刻被电光照亮的他的身姿，千万年后仍凝固在传说之中。
> 若天压我，劈开那天，若地拘我，踏碎那地，我等生来自由身，谁敢高高在上？

《悟空传》是一个时代的传奇，多年以后，相信仍然有人记得那个苦苦对抗宿命的猴子。也许我们每个人内心深处都是个无法无天的石猴，充满愤怒和激情，只是随着年龄的增长慢慢被戴上了紧箍咒，然后用一生去寻找自我。

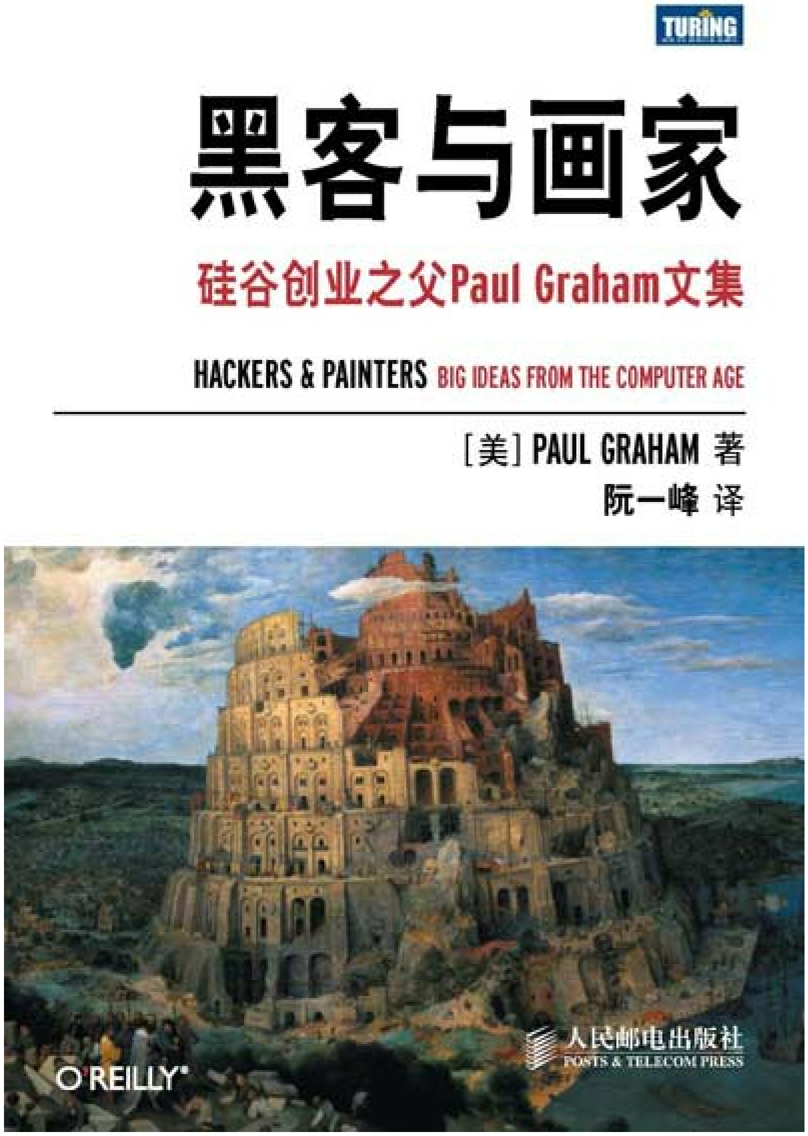
重读黑客与画家

【发布日期　2013年3月1日】

《黑客与画家》这本书的中译本出版于2011年4月，它的作者是美国互联网界"创业教父"，哈佛大学计算机博士Paul Graham，他的译者是著名blogger、译者阮一峰先生。

这本书在2011年一上市就受到了广大人民群众的爱戴，我在第一时间拿到纸质书后，通读了两遍，当时感觉很震撼，可以说本书是我近年来读过的最优秀的人文类技术图书，个人非常喜欢，所以在去年在图灵推出多看电子书后，又购买了电子版《黑客与画家》，放到手机和Pad上随翻随看，最近又开始重读。

好书的特点是常读常新，一本好书往往沉淀了作者几十年的岁月，绝不是你草草翻上一遍就可以理解和掌握的。虽然是同样的文字，在不同的时间和环境阅读，往往给你带来不同的启示和感悟。下面是我重读的读书笔记。



Redesign——设计永无止歇

Paul Graham在黑客的第十四章-梦寐以求的编程语言里，写了一节关于Redesign的随笔。什么是好的文字？好的文字来自于不停的修改，好的编程语言和软件产品同样如此，在个人的工作生涯里，我的体会是，再多的修改也不过分。可以说没什么软件产品是完美的，完美主义者都是不断打磨产品以趋近完美。如果不信，那么各位看官可以打开你们一年前写的代码或文章，如果脸红的话就吱一声吧。

同样，如果你想不断的调整自己的设计和实现，那你就需要保证你

的工作在某个特定阶段是可持续的。我的建议是无论选择公司，还是在公司内部选择工作，尽可能选择能够长期投入和完善的事情做，如果你半年做一个项目，之后又换成另一个，然后再换，除了积累了一大堆项目经验之外，1、你个人能力没有得到提升，2、你永远无法完成一个优秀的产品。在产品公司，你可以为优化某个算法或Ajax效果花费一个月的时间，这在以单纯做项目或外包的公司是不可想象的。找一家程序员被当做天才和宝贝的公司，去做可持续的产品……

推荐两段核心文字，大家体会下阮一峰老师的翻译功底：

为了写出优秀软件，你必须同时具备两种互相冲突的信念。一方面，你要像初生牛犊一样，对自己的能力信心万丈；另一方面，你又要像历经沧桑的老人一样，对自己的能力抱着怀疑态度。在你的大脑中，有一个声音说“千难万险只等闲”，还有一个声音却说“早岁哪知世事艰”。(You have to be able to think how hard can it be? with one half of your brain while thinking it will never work with the other.)

如果你能平衡好希望和担忧，它们就会推动项目前进，就像自行车在保持平衡中前进一样。在创新活动的第一阶段，你不知疲倦地猛攻某个难题，自信一定能够解决它。到了第二阶段，你在清晨的寒风中看到自己已经完成的部分，清楚地意识到存在各种各样的缺陷。此时，只要你对自己的怀疑没有超过你对自己的信心，就能够坦然接受这个半成品，心想不管多难我还是可以把剩下的部分做完。

时间与流行

一种编程语言想要变得流行，最后一关就是要经受时间的考验——《黑客与画家》。

时间有时候是很无情的，很多流行的东西，随着时间的流逝将变得面目全非。美貌与时尚如此，技术同样要经历岁月的洗礼。进入21世纪以来，技术热点不断变更，每次的技术更迭，就像流水冲刷河床一样，虚无的流走，沉淀下来的才是有价值的东西，我们要做的就是那些能沉淀下来的东西，这需要去判断和选择

进入21世纪，从静态语言到动态语言、面向对象编程到函数式编程、从模型驱动设计到领域驱动设计、从SOA到云计算、从BI到大数据、从BS到移动互联，无论技术热点如何变化，站着挣钱的永远是那些踏踏实实做产品和技术的。2009年SOA火热的时候，每家软件公司和互联网公司都号称自己面向服务了，甚至IBM、BEA等公司为SOA确定了SCA和SDO规范，但3年以后，无人再提SOA，无非就是用开放的技术实现原来的EAI么……

有位在校大学生问，“我们下学期要学习数据库，以后想从事大数据方向，不知道从什么方面学习大数据”。我的建议是，不要被现在大

数据的噱头忽悠了，等你两年后毕业出来，可能已经没人提大数据这回事了。如果真的对数据感兴趣，踏踏实实把关系数据库学好，有时间的话再学一门编程语言，掌握数据结构，再有余力学习一些数据挖掘和推荐算法等知识，这就行了，相信我，你没那么多时间！

所以，正在流行的东西并不一定值得投入，流行感冒倒是流行，也没人追啊……

最大化你的价值

有人问，你为什么要从事IT技术研发工作？如果是乔布斯，可能的答案是改变世界；如果是人生导师，可能的答案是跟随你心。如果是我回答呢，答案就是如果不从事这个行业的话呢，我还真不知道该如何养家糊口：）

每个人进入一个行业，有必然性也有偶然性，现在想来，我进入软件行业可能是兴趣使然。我小时候就比较喜欢电子相关的东西，这个习惯保持到现在就是喜欢电子设备。大学的时候电子和计算机相关的课程都学的很好，其他专业课则一塌糊涂。毕业后进入一个工厂做电子设备测试，晚上则用车间的电脑学习编程，很快自己觉得可以找到工作了，就去面试，一面之下进入洪恩，稀里糊涂算是进入了IT行业。

工作了十几年，发现自己确实对技术比较感兴趣，曾经有很多机会转岗成为纯粹的管理和业务人员，但最终都没舍得放弃技术，当然技术也没什么大成，爱好而已。

为什么做技术，技术是什么？我的观点是：

1、与其他工种一样，技术可以谋生。很多导师说看准一件事情就全情投入，不要考虑收入，奋斗不息财富会随之而来。且不说只有你成事了财富才能来，就说没成事的时候我们总不能饿得头昏眼花去奋斗吧。准确一点是工作初期不要过分考虑金钱。总要解决温饱问题吧，那么做技术研发可以很容易达到这个目标。

2、技术是一种手段和做事方式。尤其是在现在这样一个互联网和数据的时代，可以说技术面前人人平等。你付出了多少，差不多就会得到多少。很多人羡慕创业公司的人获得的财富，他们只不过是把你20年平稳的打工生涯压缩成4年艰苦卓绝的创业，当他们冒着成为炮灰的风险在清晨的寒风里编程时，你正在温暖的被窝里做着美梦。所以就别羡慕了，那是他们应得的。

3、做技术需要终身学习，如果你个学习狂，恭喜你找到了一份完美的工作。有一次一个工程师告诉我，每次感到恐慌的时候，就开始学

习，掌握了一门又一门语言和技术。最后他成为了一个通才之上的专才。书到用时方恨少，事非经过不知难，有时间就学点东西，没坏处，还能预防老年痴呆。

4、作技术可以最大化你的价值。如果你是卖煎饼的，卖一个是一个，如果你开发了一个千万人使用的软件，那你做这个软件的价值就放大了千万倍。如果你做的互联网产品服务了千万个用户，你做的这个产品的价值也就放大了千万倍。如果你在做这样的工作，那么你就最大化了自己的价值，财富也会随之而来。如果没有，就去找这样的工作。

难易相成

古人云：有无相生，难易相成，长短相形，高下相倾。

很早就想说说困难和容易这点事。人生一世，我们到底是要选择做容易的事还是做难事？这个想法源于春节前的业务讨论会，有两位经理各执一词，一个说，为了让部门能够生存下去，我尽可能去做那些简单的相对稳定的任务，因为这样可以保证良好的现金流。另一个说，为了能让部门生存下去，我尽可能去做那些困难的利润更高的任务，因为这样可以保证良好的现金流。

难易相成，困难和容易会在不同的环境下会相互转化，我觉得这两位经理在特定场景下说得都没错。

就个人而言，我觉得我们应该尽可能做那些困难的事情，让别人变得不那么困难。无论是做软件还是做互联网服务，其实终极意义就是你做出来的东西能否解决用户的问题，如果这是一个容易解决的问题，那么很多人早已经解决了；如果这是个困难的问题，那就意味着很多坑等着要埋你。咋选呢，似乎怎么选都是炮灰，权衡一下，选择前者基本是无用功，那我们只好选择后者，我们本身也是爱挖坑的人，况且困难面前人人平等么。

JetBrains是一家捷克的软件公司，他们做的事情就是为Java，Objective-C，Python，Ruby，JS等语言写开发IDE，给程序员写工具可不是闹着玩的，他们对IDE的挑剔基本比肩女性对化妆品，但JetBrains开发出来的工具深受程序员喜爱，各种智能，各种效率，他们获得了极大的成功，为啥，因为做着别人很难做成的事。

让自己困难点，让别人容易点，说了半天，好像就这么点事。Paul在黑客画家里提到选择哪种技术和语言去实现软件的时候，同样选择了那些困难的有竞争力的事情去做。

逃离舒适区

本来想起名为逃离技术舒适区，后来想想这个话题具备普适性，就把技术去掉了。上一节谈的是困难和容易的取舍，这一节的话题算个延续。

什么是舒适区？如果你是个新手，你就没什么舒适区，什么都不懂嘛舒适个毛啊，在磕磕绊绊的学习中懵懂前行，期间可能还伴随着老鸟的嘲笑和进度的压力，终于有一天你武功大成，乾坤大挪移练到了第五重，工作中开始得心应手游刃有余，不断有新人或老人来找你解决问题，你微笑着迎接各种挑战，淡淡的送走困难，你挥一挥手，不带走一片云彩，这是什么境界？这就是你的舒适区，这和靠在沙发上看电视的舒适不是一回事，通常进入舒适区需要花费你很多的时间和精力，需要你不断的练习，一旦进入，你会enjoy it！

这时候，如果有人胆敢让你脱离舒适区，可算要了亲命了，你会勃然大怒，轻则争吵，重则离职。这种事遇到太多了，一个写前端的你让他学习一些后端技术，一个写Java的你让他学习一下C，得到的答复可能会，Sorry, I feel very uncomfortable!

没有人学新东西的时候非常舒服，一旦经历过从新人到老鸟的过程，再让你进入陌生的领域，那种痛苦会让你自发的去抗拒。但是一个人不可能永远躲在舒适区里，逃离舒适区会有助于你从不同的角度看问题，视野会更加开阔。人总要往前走的。

就我个人而言，我有多年的vim使用经验，目前在学习emacs，我从Linux转到Windows再转到Mac，学习了多种编程语言，一直在练习自己的演讲能力并做了很多公开演讲。有时候做很多事就是为了挠自己痒处，避免在舒适区待太久。

很多人在某个地方待久了就会非常懈怠，没退休就像在养老，这时候我就知道，他们在舒适区太久了，与在哪个地方无关。

创造财富

《黑客与画家》里有两章是描述财富的，如果你是个财迷，那么就该去读读，如果你不是，读读我的读后感就行了。

财富和钱从来就不是一码事，所以大部分人只能去创造财富，因为你不是印钞机。但钱毕竟是流动的财富，所以大部分人还得通过创造财富去挣钱，那么一辈子挣多少钱合适呢？

在很多年以前，100万还能在北京买一套房子，大家还没有听说过PM2.5这个术语，蓝天还不是那么得稀有，有人写过一篇文章，大意就是一个家庭要多少钱才能正常的生活（什么是正常？好吧，你就当及格线60分理解），作者从购房、购车、赡养父母、教育子女、家庭生活、休闲娱乐、养老、货币贬值、物价飞涨等各个方面算了一笔帐，这个数字是600万。

这位同学你不用站起来了，我知道现在100万在北京只能买个洗手间，我说得重点不是这个。而且要算也很简单么，如果你觉得现在北京买套房需要400万，那就加上300，如果在老家50万就能买，减去50即可，不离谱。

如何去挣这个抽象的600万呢？如果按照60岁退休计算，大部分人要工作40年左右，大家可以计算下，如果妥妥的工作了40年，你的平均工资要达到多少才能过正常生活。当然你也可以选择创业，高风险高回报，一旦创业成功，你就摆脱了这个“正常生活”的羁绊，可以有更多的时间和空间做自己更喜欢的事。但是这些成功的创业者不会就此不再工作了，他们可能会比普通打工者工作更长的时间。

为毛？很多有钱人完全不必再工作，但是他们工作起来比普通人还欢实。不是天生受虐狂和社会压力，而是无所事事使人感到孤独和消沉。我有个朋友移民国外，由于有房产完全不必工作，天天烧烤钓鱼，半年以后，他在MSN上告诉我“看到烧烤钓鱼就想吐，我必须要找个工作了”。

另外，如果你不是银行劫匪或公务员，你应该知道财富是创造出来的，而不是抢来的。乔布斯和沃兹创建了苹果公司，为自己、员工、开发者和社会带来了巨大财富，但别人并未因此而变得贫穷。所以建议大家尽可能去做创造财富的工作，而不是掠夺财富。

另外，如果没人给你报酬，你是否愿意去工作呢？唯一可能的就是这件事比较有趣，比如Linus Torvalds免费写了著名的操作系统Linux，他当时可没想着用这个操作系统赚钱。那我为什么写博客呢？嗯，思考中……

关于这个话题，Paul表现出了很多技术之外的Political智慧，比如允许赚到大钱的人保留自己的财富，比如藏富于民：

一旦自己的财产有了保证，那些想致富的人就会愿意去创造财富，而不是去偷窃。由此导致的新技术会很容易被转化成财富。

要鼓励大家去创业。只要懂得藏富于民，国家就会变得强大。让书呆子保住他们的血汗钱，你就会无敌于天下。

完结

《黑客与画家》可以为你带来技术、生活、自由、财富等各方面的思考，确实是难得一见的技术图书。事实上这次重读还有很多感受没有形成文字，希望以后有机会还能继续补充这篇文章。


MacTalk

人物

传统的黑客——史蒂夫·沃兹

【发布日期　2013年6月3日】

今天和大家说说苹果的另一位创始人：史蒂夫·沃兹。



沃兹出生于1950年，根据《异类》作者马尔科姆的记述，对于计算机来说，20世纪50年代是个最为牛逼的年份，在这黄金十年里各类天才和大师呱呱坠地，开始了他们传奇的人生，史蒂夫·乔布斯生于1955，比尔·盖茨1955，保罗·艾伦1953，史蒂夫·鲍尔默（微软CEO）1956，埃里克·施密特（原Google CEO）1955，比尔·乔伊（Sun创始人）1954，当然，还包括史蒂夫·沃兹。（仨史蒂夫，俩比尔）

沃兹和乔布斯不同，乔布斯是纯草根，沃兹则是高知子弟（类似高晓松），老爹是忠诚的美帝知识分子，NASA航天局搞火箭的。沃兹之所以在硬件领域有极深的造诣，是因为乔布斯还没出生的时候，沃兹的老爹已经开始教授沃兹电子学知识；乔布斯在玩尿泥的时候，沃兹已经开始理解原子、电子、中子和质子。所以，哪个时代都得拼爹，差别就这么大。

由于沃兹从小就打下了良好的电子学的基础，一旦计算机大潮来临，他就顺理成章的走上计算机世界的康庄大道，进入了惠普公司谋得一份有趣的电子工程师的职位，如果没有遇到乔布斯，他想象自己的下半生都会在惠普度过，也许整个计算机历史就改写了。然而往事如风，无论有多少种选择，最终的结果就是上帝让乔布斯和沃兹迎头撞在一起，并产生了最大的化学反应。

他们在一起参加黑客聚会，搞各种恶作剧，一起制作电子设备，盗打电话，制作游戏机等等，说是一起，但是设计电路和编程的活儿基本都是由沃兹搞定，乔布斯就负责买披萨和焊板子。其实纵观沃兹和乔布斯合作的经历，基本上就是一个产品经理虐待程序员的悲惨案例，大家可以脑补以下场景：

乔布斯兴冲冲的从外面冲进来，对沃兹说，『兄弟，咱们得做个XX，你是最佳人选，只要你能做出来，咱就发达了』

善良的沃兹：『没问题，我喜欢这挑战』

『唔，我们需要在4天内完成』

『我艹，你没事吧，怎么可能？至少需要两周』

乔布斯不说话含情脉脉的望着沃兹：『You can do it！』

『好吧，十天』

『……』

『好吧，五天』

『……』

『好吧，就4天』

乔布斯吹着口哨走了，沃兹留下来干了4天4夜。与传统项目的悲惨结局不同的是，沃兹能把活儿做出来，还是世界顶级水平的，这就是天才的价值。

然后，苹果公司就诞生了，然后是AppleI和AppleII，沃兹做出了当时那个时代最伟大的个人计算机。之后苹果就上市了，那一年沃兹30岁。

在苹果公司上市的那一年，沃兹考取了飞行执照，半年后沃兹飞行时发生意外，清醒后呈现失忆状态，直到五个星期后才逐渐恢复记忆。之后慢慢淡出苹果公司，开始享受自己的人生，搞遥控器，音乐节，教育，慈善等等。而他的伙伴乔布斯则继续自己的传奇之旅。

纵观计算机的发展史，有些人在软件方面的有突出贡献，有些人是硬件技术专家，而同时擅长软硬件技术的则凤毛麟角，沃兹是其中之一。1976年，世界上只有少数人了解操作系统、存储、芯片、电路板、布线等知识，而沃兹基本上以一己之力，单枪匹马做出了AppleI和

AppleII，另一位创始人乔布斯凭借着自己出众的市场感觉和现实扭曲力场，让沃兹的作品震惊了世界。

回顾苹果公司的历史，沃兹研制的AppleII居功至伟，AppleII的销售和市场直接导致苹果公司上市，同时为苹果公司、乔布斯和沃兹积累了巨额财富。以至于乔布斯在出走苹果之后可以手握巨额启动资金来试验自己的各种想法。而且沃兹这个人非常厚道，在职期间还发起过"沃兹计划"，把自己的股权分出一部分给他觉得优秀的员工，在当时能做到这点真是让人无比钦佩。

沃兹是一个充满人文关怀的艺术家、技术天才，遵循了传统的黑客文化。沃兹的最初给自己的定位就是去做一些酷的东西，成为一个伟大的工程师，写出匪夷所思的代码，并且跟每个人都成为朋友。这样的沃兹肯定是人见人爱花见花开，但他断然不会去主动改变世界。但真实的情况是，没有他，世界也不会是这个样子。30多年来，世界变了，乔布斯变了，盖茨变了，不变的是沃兹。

用《我是沃兹》这本书里的一段话结束今天的话题：

苹果的创意来自一对生死与共的好兄弟，其中之一非常成功，他将毕生致力于创建伟大的公司，保证盈利，整合科技与人文，而另一个人则言谈幽默，对一些小玩意感兴趣，热爱技术，他在世界里挖掘趣闻，此生只为寻找欢笑。

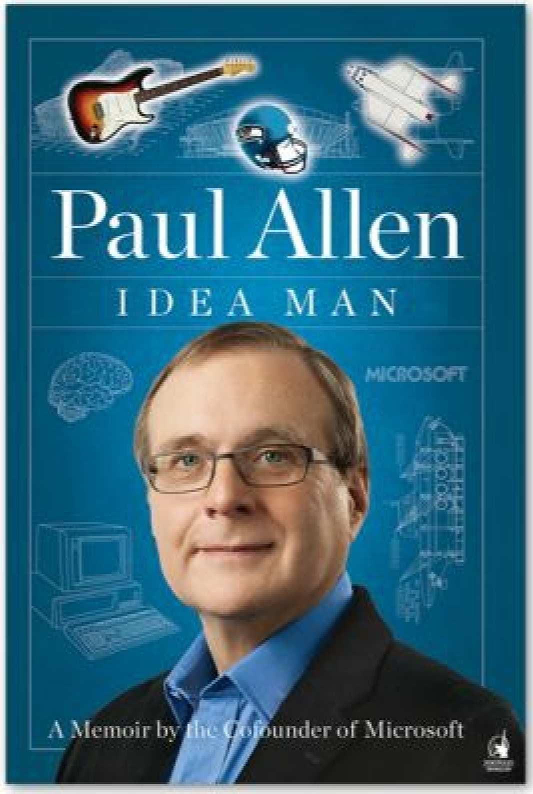
从汇编到太空——保罗·艾伦

【发布日期　2013年6月18日】

今天和大家说说微软的联合创始人保罗·艾伦吧。自从微软帝国统治了个人电脑的操作系统之后，比尔·盖茨的名声就遍及天下，我记得在2000年以后到处都是比尔·盖茨的传记在飞，但是另一位创始人保罗·艾伦就少有人提了，真正了解艾伦生平的就更加少之又少。有时候我觉得艾伦之于微软和盖茨，就像沃兹之于苹果和乔布斯，艾伦和沃兹都是不世出的技术天才，而且在公司初创和发展的过程中居功至伟，也都是在公司高速成长期间离开，去寻找其他的光辉与梦想，而他们的伙伴则继续着IT世界的传奇！

有时候看看这些技术天才的所作所为所想所言，会让人感慨万千，文艺的说法就是“感受生命的恢宏与奇迹”，通俗的说法就是“人比人得死，货比货得扔”，今天你们也感受一下！

保罗·艾伦同样出生在那个传奇年代——20世纪50年代，生于1953年，比盖茨大两岁。很多人说比尔·盖茨白手起家创立微软，好像人家是个吊丝一样，其实盖茨是标准的富二代，家境显赫，老爹是州律师协会的主席，有大House，家里定阅的杂志是《财富》，以至于艾伦第一次去盖茨的家时“感觉稍稍有些敬畏”。艾伦就差一些，老妈是教师，老爹打过二战，退伍后在图书馆工作，算个普通人家，这对父母怎么也没想到自己的儿子会在几十年后长期霸在福布斯财富排行榜的前列！



艾伦的老爹虽然是个普通人，但是作为父亲基本做到了伟岸深邃，他在艾伦很小的时候就告诉他“等你长大了，要做喜欢做的事情，无论做什么，喜欢就好”，这位老爹为什么有这样的警世名言呢？因为老爷子当年打完二战退伍之后，有两个选择，一个是做橄榄球教练，一个是图书管理员，结果一哆嗦选了稳妥的图书管理员，一辈子徜徉在知识的海量里，后悔一辈子！

各位凡事选择稳妥的童靴们感受下吧，我只能帮你们到这里了！

和史蒂夫·沃兹一样，艾伦小时候非常调皮，爱好是恶作剧，喜欢电子学，自学能力超强。老爹老妈一看，孩子有出息啊，砸锅卖铁让他上了私立学校——著名的“湖畔中学”，在那里他遇到了身材瘦长满脸雀斑的比尔·盖茨，并由于共同的爱好和革#命信#仰走到了一起，由于湖畔中学良好的师资力量和硬件设施、开放的学习氛围，艾伦和盖茨得到了自由成长。艾伦开始学习汇编语言，他读了《汇编程序手册》《系统参考手册》《操作系统手册》，在学习了几个月后对自己说“这东西太迷人了！”，这不是天生码农是神马？

一位同学说“保罗能像别人读小说那样读汇编程序”，但真实的情况是，艾伦也不觉得汇编容易，只是他比较专注罢了。

在那段时间里，艾伦和盖茨开始疯狂的编程，晚上11点开始达到巅

峰状态，并且保持很长时间，不管多晚，都要找到最后一个bug。他们经常一忙就是几天，然后睡上十几个小时，起身继续工作。据艾伦描述，比尔·盖茨的工作状态是这样的："他一会踱步，一会坐在椅子上摇晃，在黄色的纸片上胡乱记着什么，等我的模拟器完成后，他就挪到终端跟前，一边看着纸片，一边用他疾风一般奇怪的指法编写代码，接着重复以上过程，他会一连好几个小时保持这个状态"。

而艾伦已经熟悉10种类型的计算机、10种高级语言、9种机器语言、3种操作系统。那一年他不到20岁。

关于这些技术人物传记，一本书也写不完，我只是写一点有趣的、能让我思考的东西，如果想了解他们的传奇经历，还是去读个人传记和回忆录更好一些。

伟大的梦想都始于微不足道，当年艾伦和盖茨在畅想创业生活的时候，艾伦问比尔，"如果一切顺利的话，你觉得我们公司能办多大？"比尔想了想说，"估计得有35名程序员那么大！"，艾伦心想，这特么真是个雄心壮志！

1975年，艾伦和比尔承接了MITS公司的一个单子，为牵牛星电脑编写Basic语言，他们开始了进入新一轮的编程竞赛。工作强度之大难以想象，以至于在去餐厅吃饭的时候经常会吓到餐厅的服务员。人家看着他们一个个脸色苍白，小声问"你们是不是在飙车？""没车彪毛啊，哥在编程！"

他们要么噼噼啪啪的敲击键盘，要么大口吃饭，要么蜷在办公室的地板上睡的不醒人事。用一句话描述就是：写不完的代码、改不尽的bug和永远不够的时间。

终于代码写的差不多了，人家要和他们签合同（现在企业客户经常让软件厂商先干活后结算，木想到米帝70年代就这样了），这时就必须有个公司，比尔和艾伦想了半天，最后艾伦想出了"微软"这个名字，然后就是股权问题，艾伦想当然的认为该五五开，这时盖茨开始语重心长的对艾伦说："你拿一半不合适，你看，你是兼职，还有薪水，我毛都没有；我编程和睡觉时间都比你多；我的代码比你多，我的bug比你多，balabala……，所以我应该比你多，六#四吧，就这么定了"，艾伦听完以后觉得太特么有道理了，于是就六#四了，当然后来艾伦搞清楚局势之后想反悔已经来不及了，很多人说乔布斯的扭曲现实能力，我觉得盖茨在这方面也毫不逊色：）

当然，即使是40％，微软在艾伦离开之后的爆发式成长一直源源不断的为艾伦提供财富，估计有一段时间艾伦每天睡醒之后就是查看自己

的户头又多了几千万美金，我要说这特么也是个乐子。

当然，艾伦毫不质疑比尔·盖茨的编程能力，盖茨当年要编写软盘驱动器的Basic程序，还有几天的时候抓着三个作业本和十根铅笔住进了旅馆，5天后带着几千个字节的汇编代码回来了，然后满脸倦容把这些代码输入终端，告诉艾伦“搞定，你们再看看有没有bug”，然后就回哈佛了。

牛逼的人就是这样工作的，搞定之后飘然离去，不带走一个字节！艾伦当时的评价是：随着公司发展和成倍的管理职责，比尔很少有机会做这样高强度的程序创作，在编程上比尔有无与伦比的才华！

之后，盖茨从哈佛退学，两个人开始专心经营微软公司，后面的事情大部分搞计算机的都比较清楚，IBM公司选中微软公司为其编写新PC的操作系统，MS-DOS诞生，微软开始头角峥嵘，迈上发展的快车道后一发不可收拾，直到成为全球最大的软件公司。然而，艾伦和比尔这对好基友的关系却发生了一些变化，两个人开始意见相左，而盖茨更强势一些。1982年艾伦患了了一种叫做霍奇金的疾病，心力憔悴，逐渐退出了微软，但他依然是微软的第二大股东。

艾伦退出微软之后4年，1986微软上市，市值达到5亿美金，盖茨和艾伦瞬间成为亿万富豪，从此两个人也走上了完全不同的道路。艾伦手握微软的巨额资金，不断的更新自己的兴趣和梦想，他之后的一系列举措是：

* 买下了NBA开拓者队，进军NBA。
* 买下了西雅图海鹰队（橄榄球），打进超级碗比赛。
* 出资修建一座称为体验音乐计划的交互式音乐博物馆，成立乐队，负责节奏吉他和主音。
* 赞助太空飞行计划，准备上天看看，失败，身体不好有钱也木用啊。
* 建立艾伦脑科学研究所，研究人脑图谱，试图解开大脑的奥秘。
* 54岁高龄去南极冒险，看看企鹅看看冰。
* 60岁，写了一本书叫做《Idea Man》，那位同学请注意了，这特么不是钢铁侠，中文名叫《我用微软改变世界》。
* 一直以来做慈善事业，拟死后向慈善事业捐献大部分财产。

精彩人生继续……

保罗·艾伦与比尔·盖茨共同创建微软公司，艾伦8年后离开，凭借在微软的股份先后涉足体育、音乐、太空、人脑等多项领域。而比尔几十

年来一直强力经营这世界上最大的软件公司并成为世界首富，之后退休做慈善。这两种生活估计是很多人的梦想，各位看官，你们梦想哪种生活呢？

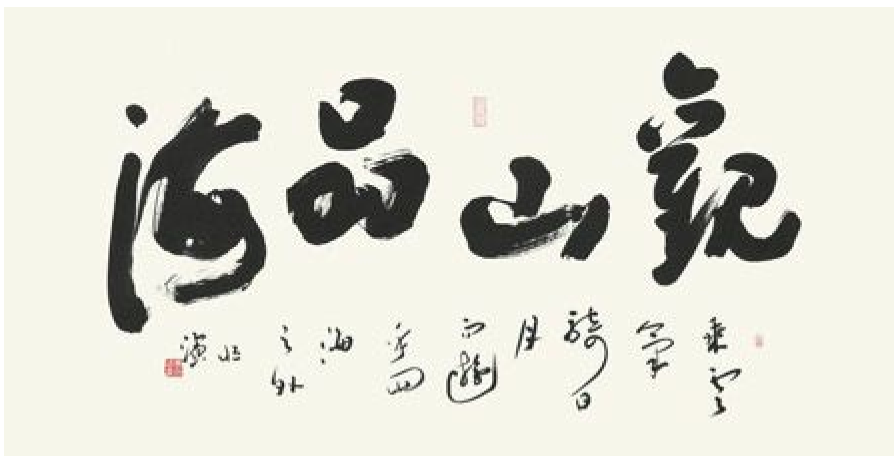
另外，我读过很多技术天才的传记，除了天分以外，他们都是极其勤奋，并痴迷于自己的事业，不眠不休。估计这些牛人在专业技能的钻研上全部都超过了一万小时。要想人前显贵，人后必须受罪，人后若不受罪，人前就要遭罪，你们感受下，Mac君真的只能帮到这里了。

敬畏之心

【发布日期　2013年8月8日】

最近在读两个人的著作，一个是冯唐，一个是吴军。

喜爱文学的人应该对冯唐不陌生，当年韩方大战时，冯唐及时抛出文学金线论，后被戏称为“冯金线”。古诗有云，“冯唐易老，李广难封”，冯唐当然是他的笔名，此人真名“张海鹏”，似乎比Mac君的名字还要俗气一些。



冯唐是协和妇科博士，前麦肯锡合伙人，作家。我最初知道冯唐的时候，被丫的三重身份惊呆了，这得多分裂才能把这三个职业捏合到一个人身上？比无间道牛，比韩寒的赛车手和作家身法高端大气。

冯唐的文字有幼功，初中的时候熟读大量的古文和外语著作，所以他的文字非常体现功力，能少用一词，绝不多填一字，文字充满速度而不凝滞，厚重而有质感。文学的幼功咱不懂，围棋的幼功我是知道的，我哥哥半路出家学习围棋，打败同时代学期小伙伴无敌手，但是经常被那些三四岁开始摸棋的小屁孩打得落花流水而无还手之力，我问何解？我哥说这是幼功！可见一斑。

冯唐写过很多书，我最近在读的是《三十六大》，是他今年的杂文集，以冯唐的才气，写杂文当属大理石压咸菜缸——大材小用，但是这些文字依然能够体现他的风格和特点，我非常喜欢，摘录几则供大家欣赏：

个人和全体古人的关系，应该是昆仑山上一棵草和昆仑山的关系。在长出草之前，需要先

爬昆仑山。如果不明白什么叫高山仰止，先别说“俱往矣”，先背三百首唐诗。知道昆仑山有高度之后，开始爬吧，学杜甫学到风雨掀翻你家屋顶，学李白学到梦里仙人摸你头顶，学李商隐学到你听到锦瑟的一刹那裤裆里铁硬。学到神似之后，是血战古人，当你感觉到不是自己像杜甫、李白、李商隐，而是杜甫、李白、李商隐像自己，就是到了昆仑山顶。是时候长自己的草了，不是杜甫的草，不是李白的草，是自己的草。这个时候，长一寸，也是把昆仑山增高一寸，也比自己在平地蹦跶一米，高万丈，强百倍。

关于现场，你说：“笔与墨会，是为絪缊，絪缊不分，是为混沌……不可雕凿，不可板腐，不可沉泥，不可牵连，不可脱节，不可无理。在于墨海中立定精神，笔锋下决出生活，尺幅上换去毛骨，混沌里放出光明。纵使笔不笔，墨不墨，画不画，自有我在……人写树叶苔色，有深墨浓墨，成分字、个字、一字、品字、厶字，以至攒三聚五，梧叶、松叶、柏叶、柳叶等垂头、斜头诸叶，而形容树木山色、风神态度。吾则不然。点有风雪雨晴四时得宜点，有反正阴阳衬贴点，有夹水夹墨一气混杂点，有含苞藻丝缨络连牵点，有空空阔阔干燥没味点，有有墨无墨飞白如烟点，有如胶似漆邋遢透明点。更有两点，未肯向学人道破，有没天没地当头劈面点，有千岩万壑明净无一点。噫！法无定相，气概成章耳。”现场有神。

无论是钢笔在纸面上书写还是手指在键盘上敲打，我知道，字句的黑白疏密凸凹之间，有小鱼和小雀在。

两千多年前，人平均寿命不到五十，孔丘说，一个人到了四十，知道了自己能力的边界，能做什么和不能做什么，于是不惑。两千多年后，人平均寿命超过七十，孔丘说的依旧适用，这个老怪物。这几天冬去春来，换季节，睡得不安稳，昨夜醒来，看到你就倚在窗台边抽烟，生命就像一头驴一样蹲在你旁边，因为彼此熟悉、天人相知，驴血已经不嗞嗞作响。一时，我想，我想骑就骑，要下就下，打打小鸟、看看小星、码码小说，向死骑去而不知死之将至，一切挺好。

冯唐还有一本奇书叫做《不二》，国内是没有出版的，如果你不知道，就用Google百度一下，我觉得冯唐的《不二》让他在汉语文字的使用上抵达了又一个巅峰，我明显能感觉到他在尝试汉语使用的极限，在用文字打败时间！500年后，文史应有《不二》的一席之地。

说了冯唐，有些读者表示不喜欢冯唐，有的说看不懂，有的说Mac君你怎么能喜欢冯唐呢？声明一下，Mac君不喜欢冯唐，俺是异性恋，只是喜欢冯唐的一些文章和文字罢了。另外，写冯唐并不是推荐大家去买或去读冯唐的书，只是主观表达，文学这东西，确是仁者见仁淫者见淫，萝卜白菜各有所爱，大家只管去读自己喜欢的书就好了，多读书，是好事。

下面说说吴军博士和他的书。吴军应该算是科学家范畴的，毕业于清华大学和约翰·霍普金斯大学，获博士学位，主要研究领域是语音识别和统计语言模型，和李开复老师的早年研究方向类似，当然李老师…咳咳，已经不做技术和研究很多年了，但吴军博士依然是冲在第一线的，他先后加入Google和腾讯，都是负责搜索领域的研究和研发工作。

吴军目前出了两本书，分别是《浪潮之巅》和《数学之美》，原稿都来自他在Google黑板报的系列文章。很多人知道吴军可能是因为《浪潮之巅》，那本书我当然也读过，目前在读的是《数学之美》。

我以前说过，很多人技术做的很广很深，但是文字能力欠缺，写出东西来让读者痛苦不堪；有的文字功底深厚，但技术不行，著作虽然读来容易，但无法深入浅出，读者一旦醒悟过来后果同样很严重。二者兼而有之的人则少之又少，所以优秀的技术图书真是凤毛麟角。吴军在技术和文字层面都属上上之选。

《浪潮之巅》出版之后赞誉多多，我读完以后感觉比那本《硅谷之光》好看，有些人觉得有不符史实之处，但我觉得吴军凭一己之力纵论IT科技和公司发展史，揭示规律品评趋势，书写科技公司的人、事、势，同时辅以自己的思考和论证，确属难能可贵，行文流畅，读来收获多多。

今天主要说说我正在读的《数学之美》。小时候念书时，数学是我的强科，在几门学科里处领先之列，但工作以后除了记得几个算法和编程所需的一些数学知识之外，其他大部分都还给老师了，想想老师也不易，那么多同学那么多知识还回来，得有多沉多重。知识虽然还了，但兴趣还在，常读一些数学相关的著作，这本《数学之美》很早就买了，但近期才拿起来看，说来也是惭愧的紧。

数学是众多自然科学、社会科学、管理科学的基础，誉为是科技之根毫不为过。《从一到无穷大》一书在第一章“大数”里讲了个故事，说两个贵族决定做计数游戏，谁说出的数大谁赢，

一个贵族绞尽脑汁想到了一个大数：

“三”（请以shan平音读出并感受下……）

另一个贵族傻眼了，想了想说，“好吧你赢了”。

这当然是个故事，用来说明原始社会物质匮乏导致无法产生大数，吴军在《数学之美》里也引述了这个故事，并由此开始讲述数学之美。整个书的基调是简洁、扎实、深入浅出又栩栩如生，于平凡之中见深邃，于无声处闻惊雷，读来有趣而又涨姿势。对于从事IT技术的人来说，以下内容值得细细研读：

* 自然语言处理
* 统计语言模型
* 中文分词
* 搜索引擎的索引
* 图论
* 网络爬虫
* PageRank网页排名
* 地图和本地搜索

* 拼音输入法的数学原理
* 逻辑回归和搜索广告……

这本书读起来是比较费力的，不跟看网络小说那样一个晚上翻250页，但费力之事自有其好处，人生总要做一些艰难费力的事情，读书也是一样。

写到这儿很多读者要问了，这和敬畏之心有啥关系？这是近期Mac君在读这些书的时候产生的一些感受。很多读者看到我写了一些文和事，经常发信息说好说赞，但我自己经常处于一种惶恐的状态，比如读书的时候，你就会知道自己文不如冯唐、术不及吴军，其实用不如是不妥的，实在是难以望其项背。一个人学了十年再工作十年，以为学了一点文和术，但你了解愈多，会发现自己仅仅是大海之一滴水、沙漠之一粒石，永远有无数优秀和卓越的人在你前方奔跑，你只能看到他们奔跑中的一缕尘埃！

但在现实中，却是敬畏之心不常有，常起轻视之意。所谓文人相轻，自古如此，大家执笔对骂，恨不得在对方身上戳几个窟窿。程序员也是一个德行，看见别人的代码要么是Garbage，要么就Refactoring，缺乏起码的尊重和敬畏。总之文章和代码都是自己的好，LD都是别人的好（LD可做多重含义理解）。

以前说过，人的一生要么就低估自己，要么就高估自己，想要做到既不妄自尊大也不妄自菲薄，实在是太难了！我们只能保持敬畏之心，保持孤独之意，一路向前！

设计巨匠——乔纳森·艾维（**Jonathan Ive**）

【发布日期　2013年5月29日】

今天看到一篇新闻，说苹果的CEO库克参加《华尔街日报》主办的D11科技大会时，谈了一些大家关心的热点问题，包括对Google Glass的看法，苹果的可穿戴技术策略（iWatch？），iOS平台为第三方开放更多的API，乔纳森·艾维主导iOS人机界面的设计等等。



Google的眼镜和Apple的手表离我这样的土老冒比较远，即使今秋发布iWatch，估计我也就是擦擦口水看一眼的份，第三方API呢，有新的用就是了。但是iOS或OS X的升级是关系到“果计民生”的大事，所以我对最后一点比较感兴趣，其实也是对乔纳森·艾维这个人比较感兴趣。

艾维是哪路神仙呢？对于大部分中国人民来说，我们熟知的是苹果的灵魂人物和精神领袖乔布斯，如果再知道一个联合创始人沃兹就算逆天了，但是沃兹很早就离开了苹果，开始享受人生。真正与乔布斯并肩战斗并实现苹果伟大复兴的，是艾维。在苹果，这兄弟基本上是乔布斯的精神伴侣，据说乔布斯在公司吃饭的时候，很少有人凑上去找骂，只

有艾维能过去陪吃陪聊。

从设计成就上，乔纳森·艾维负责了iPod系列、iPhone系列、iPad系列的、iMac系列的产品工业设计和人机交互设计。这些设计为他带来的荣誉是：

2005年被英国女王授予“最优秀不列颠帝国勋章司令官”

2007年因为iPhone的设计获得美国国家设计奖

2010年《财富》杂志将其评选为『世界上最聪明的设计师』

你们想一想，这个人得多牛逼啊，而且最难得的是牛就牛吧，还低调的令人发指。在乔哥辞世之前，几乎不接受采访，很少做公开演讲，偶尔当一下乔布斯的串场嘉宾，接个电话唔得。但是因为长得帅，发布新产品的时候总是被选为产品代理人。一般效果是这样的，在苹果传统的纯白背景中，虎头虎脑的艾维冒出来，苦口婆心的告诉你，兄弟今年可没歇着，又做了一款这个星球上最牛逼的产品，你们看看，这，这，这，amazing，fantastic，好了，钱包拿来balabala……非常有意思。我其实就是从苹果的视频广告里知道这位设计奇才的。

另一件有意思的事就是，我一看到库克那张方脸，就觉得丫是搞库房管理的，再看人家艾维，从那个角度看都充满艺术感，所以说，如果你们没有设计出牛逼的产品，可以对着镜子评估一下“哈哈”。

乔纳森·艾维是一位英国人，在乔布斯1997年回归苹果之前就已经在公司负责一些产品的设计。据说在乔布斯回归的时候，艾维正准备收拾东西走人，所幸乔布斯发现了艾维和艾维的设计工作室，我琢磨当时乔布斯的想法是：“我嚓，捡到宝贝了”。从此两人开始天作之合的设计旅程，第一个产品就是彩色透明的iMac，然后一发不可收拾，iPod的转轮，macbook的铝合金设计，iPhone、iPad、MacAir等等。

如今，这个地球上最牛逼的设计师开始同时负责iOS的硬件和软件设计，这样的人机交互界面是不是值得期待呢？今年的WWDC应该会给出答案！

关于艾维，如果你觉得没看过瘾，可以去读读《乔布斯传》，其中有一章专门讲“大师艾维”。我想以后可能会有一本艾维自传问世！

MacTalk



工具

VIM——缘起

【发布日期　2013年8月1日】

从湿漉漉的广州回来，兜头遭遇一盆大雨，大雨过后，整个京城变得清凉干爽，俗话说，环保基本靠风，清洁基本靠雨，此言非虚。傍晚从办公室出来，满园的白玉簪花散发出淡淡清香，碧叶莹润，花色如玉，新鲜的雨珠还挂在绿叶和白花上，一点一点，晶莹剔透，折射着夕阳的余晖，闪闪烁烁，煞是好看。

最近不断有vim的关键词在微信后台的回复中出现，好把，今天就开始给Vim系列搬家。

操作系统、编程语言和编辑器是程序员永恒的吐槽话题，技术发展了几十年，争论起来依然是“此恨绵绵无绝期”。在文本编辑器领域，Vim和Emacs是永恒的焦点。我在“如何学习一门编程语言”里写道，“Emacs和Vim程序猿，大家沿着不同的道路和目标前进，但总是会在某个点交叉相遇，见面就扔石头和臭鸡蛋，砸得对方鼻青脸肿，然后擦擦眼泪和口水继续前行”。你们就知道，编程其实是个危险的工作，当真不是瞎扯。

工作十几年来我用过各种编程工具，用错过，也用对过，虽然每种优秀的编辑器都有传奇的故事，每个程序员都有自己的脾气，但这一次，我只想写一写Vim……



为什么Vim总会和Emacs相提并论呢？因为Vim号称编辑器之神，Emacs则是神的编辑器；Vim编程唯快不破，插件遍天下，Emacs则宣称自己是伪装成文本编辑器的操作系统，所谓针尖对麦芒，就是这样的效果，写完Vim这个系列，我希望还有机会写写Emacs。

缘起

我从2000年开始接触Vim，当时正值第一波互联网浪潮，我刚毕业不久，一如现在的热血青年，投身到互联网的大熔炉中（当然和现在的互联网弄潮儿不一样的，当年我们大部分都被熔掉了）。

当时的公司聚集了很多清华北大的兄弟，技术牛人扎堆，大家清一色使用Vim在服务器端编程，语法高亮都不设，内部BBS也是水木清华那种，通过终端访问，非常高端大气。走进办公室一眼望去，满目皆是黑漆漆的屏，绿瓦瓦的字，每个人都在那里噼噼啪啪的敲击键盘，韵律十足，我想，这简直酷毙了！

当时我还在使用Editplus编程，隶属菜鸟帮。别人的开发、编译和发布环境都在服务器端，而我则需要在本地编写好程序，通过Editplus的ftp功能上传到服务器端，再进行调试、测试和发布，非常麻烦。

所谓文人相轻，我这种行为遭到了小伙伴的无情嘲讽，当年的Mac君只好眼泪往肚子里咽，把愤怒都发泄在键盘上，每天在满天星斗的夜色中学习Vim技法，在清晨的微光中编译Linux内核，上午敲打键盘输出Perl程序，中午吃完五又四分之一口米饭之后开始研习Vim的多窗口和标签……由于单身住在公司，时间充裕到让你不好意思，很快小成，编码时鼠标锁进抽屉，双手敲击键盘上下翻飞，成就感十足，我对语法高亮情有独钟，经常把自己的界面配置的花花绿绿，没事看看也是件乐事。

自此以后，与Vim结下不解之缘，十几年过去了，工作一直没有离开过Vim，断断续续一直在用。到了2009年，我开始把工作环境完全切换到了Mac上，记得当时打开Mac的终端时，欣喜若狂的想，这不就是Vim、Shell和IDE的完美集成么？

场景

在不同的场景下采用最合适的工具，永远是最佳选择。这是就会有人问，Vim适合什么场景呢？简单说来，Vim比较适合Unix/Linux服务

器端编程，当然这因人而异。我个人使用Vim主要用来进行Shell/Python/C编程。在Unix/Linux服务器端编辑和修改文件也离不开Vim，另外由于我个人工作环境是Mac，所以改个文本文件查个代码的，也就用Vim顺手做了。

与很多程序员交流，大家会认为不用Vim一样能修改服务器端的文件，ftp拉下来，改好了再传上去。这当然是一种方案，但不是最优方案。而且极端情况下需要直接在客户服务器上解决问题，你总不能说“骚瑞，Sir，我不会在Linux下编辑文件，Down一份下来先”？这就像用Vim编写Markdown文件一样，能不能用，当然可以，甚至有人已经为Vim开发了Markdown插件，可以编写时通过快捷键查看转换的HTML文本，但是显然不如Mou/Byword。

编写JavaEE、HTML/CSS/JS、Objective-C，最优方案可能是Eclipse、IDEA、XCode等，这些优秀的工具可以帮助我们提升效率，减少错误，但是如果你还想更进一步，那么Vim绝对值得拥有。

Vim用了很多年，也总结果过很多次，但不完整，我想这次在MacTalk能有始有终，完成这个系列。网络上介绍Vim的文章浩如烟海，各种精巧的用法数不胜数，看起来还是比较费时费力。Vim本身开箱即用，什么都不配置也可以使用其基本功能，如果大家掌握了基本操作，以下内容可以帮助提升效率，打造你的专属Vim。（待续）

VIM——为效率而生

【发布日期　2013年8月4日】

现在回想早年用Vim编程的场景，当可体现“专注”二字，那个年月的网络没有这么多诱惑，编写代码时差不多只开一个终端工具，噼噼啪啪的敲击键盘，累了就躺在办公室外的沙发上休息，也可以站起身看看窗外，舒缓一下疲惫的眼光就能延展到清华东门上空。五道口还地属荒凉，没有高楼，夕阳可以直接从窗口进入室内，光影打在办公桌上让人感觉非常温暖。

时过境迁，现在靠着硬盘的快、内存的大，我常常同时开几十个程序，用command+tab一切，屏幕中央一排几毫米见方的小图标一字排开，看着它们我有时候茫然若失，不知道自己到底想点开哪个，估计它们看着我也郁闷，“这孙子把我们都打开想干什么？”环境变好，硬件充沛，软件强劲，我们的效率是下降了呢，还是下降了呢？

而Vim，正是为效率而生的。



历史

Vim源于Vi，但不是Vi，Vi作为计算机的文本编辑器历史极为悠远，它是由美国计算机科学家比尔·乔伊编写并于1976年发布的，同年苹果公司成立。我在“传统的黑客——史蒂夫·沃兹”中提到过比尔·乔伊，他是Sun公司的联合创始人和首席科学家，一位传奇的技术天才，我个人以为他最伟大的贡献是独立编写BSD操作系统，开发Vi编辑器，创立Sun公司，当然，他还是Java语言的主要贡献者之一，任何人有幸完成其中一项工作已经足以名垂计算机发展史，而乔伊则通过一己之力

完成了这些科技成果，推动了整个计算机科技的发展。

Vim则诞生的要晚一些，它的第一个版本由布莱姆·米勒在1991年发布，这个兄弟也是一位声名显赫的程序员，80年代买了一台Amiga电脑，打开电脑一看，米勒鼻子差点气歪了，居然没有他最常用的Vi编辑器！对于米勒来说这是不可接受的。

愤怒的米勒决定自己开发一个文本编辑器，完全复制Vi的功能，并起名为Vi IMitation（模拟）。事实证明，牛X的程序员都具备这种德行，感到不爽了，就会写出个什么东西，要么完善一下，要么创新一下，要么是你写，要么是我写，于是很多伟大的软件程序就发明出来了。随着Vim的不断发展，更多更好的功能被加了进来，正式名称改成了Vi IMproved（增强），也就形成了现代的Vim，目前最新的稳定版本是7.3，Vim的开发语言是C和VimScript（后续我们还会谈到这门语言）。

理念

Vim是一款完全面向程序员的软件，活了三十多年我还没有见到过用Vim编辑文字的普通用户，如果你是，你一定要告诉我。

写过程序的人都知道，编程的时候双手大部分时间都放在键盘上，或编码、或插入、或移动、或定位、或查找，这种连续操作的时间和频率远远大于阅读、翻页、设置字体、摆弄样式等文案工作，而二者往往产生很多停顿和间隙，而编程时的停顿是非常影响编程效率的，所以Vim的设计理念就是通过模式的转换、命令的组合和数以万计的插件，保证程序员在编程的过程中，双手尽可能保留在键盘中央的区域，并且，不需要鼠标。

想用好Vim，先要理解Vim的模式转换。Vim常用的模式有四种：

1．普通模式：Vim启动后的默认模式，用来移动光标、删除文本、覆盖输入文本、恢复操作、粘贴文本等等。

2．插入模式：输入i或a进入插入模式，在这个模式下敲击键盘会往文字缓冲区增加文字，相当于普通编辑器的编辑模式。

3．可视模式：选择文本，可以行选、块选和依次选择，选择后可以进行复制、删除、排序等操作。

4．命令模式：执行内部和外部命令，通过“：”“／”“？”“！”可以进入命令模式，分别对应的是：执行内部命令、向上或向下搜索、执行外部命令。

Vim的模式和普通的编辑器有所不同，而且命令繁多千变万化，所

以初期的学习曲线较高，一旦你坚持练习并且度过了最早的平台期，就会领略Vim的妙和全键盘的好。事实上Vim除了能够快速编辑文本文件之外，还能够通过简单的命令做更多的事情，所以，“如果你认为Vim只是一个文本编辑器，你就输了！”

举几个例子，大家领略一下Vim的风采：

1、如果你想在Vim打开的文本中插入一个1到100的序列，执行如下命令：

```
:r!seq 100
```

2、如果你想在当前的每一行文字前面增加“序号.”，那么执行如下命令：

```
:let i=1|g /^/s//\=i.". "/|let i+=1
```

3、如果你想把当前目录下（包括子文件夹）所有后缀为java的文件中的apache替换成eclipse，那么依次执行如下命令：在当前目录下执行：

```
vim
:n **/*.java
:argdo %s/apache/eclipse/ge |update
```

后续还会为大家介绍一些Vim使用技巧，不过这个系列不会去讲如何通过hjkl移动光标，如何块选、行选，如何查找、定位，如何跳到行首行尾等等，我会告诉大家如何通过Vim自带的帮助去学习这部分内容，当然，如果大家有需要，我可以出一个常用快捷键操作列表。

下一篇Vim文章的内容是：“使用帮助”和“环境配置”。

VIM——帮助和配置

【发布日期　2013年8月5日】

今天说说Vim的帮助和配置内容。

使用帮助

Vim本身提供了非常详细的帮助系统，初学者可以通过帮助手册学习Vim的基础内容。在Vim中输入命令`:help`，即可进入帮助页面，默认是英文帮助，如果你喜欢看中文，可以通过以下方式安装中文帮助内容：

* 下载中文帮助的[文件压缩包]
* 解压，把doc目录下的文件复制到~/.vim/doc下
* 确认在.vimrc中设置了`set helplang=cn`
* 输入命令：`help`即可进入中文帮助
* 输入命令：`help user-manual`直接进入用户手册

用户手册的界面是这样的：

```
=====================================
总览

初步知识
usr_01.txt  关于本手册
usr_02.txt  Vim 初步
usr_03.txt  移动
usr_04.txt  做小改动
usr_05.txt  选项设置
usr_06.txt  使用语法高亮
usr_07.txt  编辑多个文件
usr_08.txt  分割窗口
usr_09.txt  使用 GUI 版本
usr_10.txt  做大修改
usr_11.txt  从崩溃中恢复
usr_12.txt  小窍门

高效的编辑
usr_20.txt  快速键入命令行命令
```

如何浏览帮助呢？请牢记如下秘籍：

* 移动：使用光标键，或者用h向左，j向下，k向上，l向右。
* 退出：使用:q<Enter>。
* 跳转到一个主题：将光标置于标签（例如usr_01.txt）上然后输入CTRL-]。
* 跳回：键入CTRL-T。
* 翻页：键入CTRL-F/B

当然，大家也可以从如下网址下载用PDF版本的户手册和帮助文档：[下载]

配置

Vim以简洁的方式提供了丰富的配置功能，主要配置体系由一个文件和文件夹组成。在一台安装了Vim的OS X/Linux/Unix机器上，进入用户主目录，可以找到.vimrc文件和.vim文件夹，这就是Vim所有的配置信息。

1、.vimrc

用户目录下的.vimrc文件就是Vim针对当前用户的主配置文件，该文件不是必备的，没有的话就创建它。文件位于当前用户的主目录下，可以用`~/.vimrc`找到，Vim启动时会自动运行文件中的每条命令。

通过.vimrc我们可以为Vim进行个性化配置，包括使用方式、显示风格、编写函数和运行插件等，.vimrc中所有的命令都可以在Vim运行时通过类似：`comm args[=args1]`的方式动态运行，即时生效。



以下是一个.vimrc的样例脚本，包含了一些常用配置，后面的注释是简要说明。

*.vimrc的注释用双引号（"）表示，样例中的大括号仅表示功能区，属于注释的一部分，无其他含义

*.vimrc的配置非常丰富，可以定义各种宏、函数、插件和映射，我见过最长的.vimrc配置有1000多行，这里的示例比较简单，适合入门级用户

```
syn on                              "语法支持

common conf {{                      通用配置
set ai                              "自动缩进
set bs=2                            "在insert模式下用退格键删除
set showmatch                       ""代码匹配
set laststatus=2                    "总是显示状态行
set expandtab                       "以下三个配置配合使用，设置tab和缩进空格数
set shiftwidth=4
set tabstop=4
set cursorline                      "为光标所在行加下划线
set number                          "显示行号
set autoread                        "文件在Vim之外修改过，自动重新读入

set ignorecase                      "检索时忽略大小写
set fileencodings=uft-8,gbk         "使用utf-8或gbk打开文件
set hls                             "检索时高亮显示匹配项
set helplang=cn                     "帮助系统设置为中文
set foldmethod=syntax               "代码折叠
"}}

" conf for tabs，为标签页进行的配置，通过ctrl h/l切换标签等
let mapleader =','
nnoremap <C-l> gt
nnoremap <C-h> gT
nnoremap <leader>t :tabe<CR>

"conf for plugins {{插件相关的配置
"状态栏的配置
"powerline{
set guifont=PowerlineSymbols\for\Powerline
set nocompatible
```

```
set t_Co=256
let g:Powerline_symbols ='fancy'
"}
"pathogen是Vim用来管理插件的插件
"pathogen{
call pathogen#infect()
"}

"}}
```

2、.vim

.vim是Vim的主配置文件夹，位于当前用户的主目录下，可以用`cd ~/.vim`进入。该文件夹一般用来放置插件和相关的帮助文档，常用的目录结构包括：

```
doc           //帮助文档目录
autoload      //Vim启动时自动加载的插件目录
plugin        //插件目录，一般在使用Vim时通过命令呼出
```

当然，如果你已经安装了足够多插件，那么这个目录下就会变得五花八门，syntax、snippets、indent等文件夹都会冒出来了。一个插件所包含的文件往往会分布在多个文件夹下，管理起来比较麻烦，稍后我们会介绍两个管理插件的插件，让这个目录变得干净整洁，容易管理，这两个插件的名字叫“pathogen”和“Vundle”！

今天就到这儿，后续文章会介绍Vim的Buffer、Window和Tab，敬请期待。

神兵利器——Alfred

【发布日期　2013年7月10日】

有人的地方就有江湖，有江湖就有纷争。

很多人说我的文字风格相对轻松和温和，那是因为我很早就认识到，我们没有教育脑残和喷子的义务。在网际多年，看过太多虚拟的刀锋和鲜血，很多人被彻头彻尾的粉碎，挫骨扬灰，似乎从来没有来过这个网络，但是很快这些人就从另一个黑暗的角落爬了起来，并换上一副簇新的马甲继续战斗。所以我在网络上很少参与或挑起争端，我的文字只写给愿意读的读者。即使这样，有时你还是会遭遇一些特别轴的人，你说“Spotlight可以用多种方式快速定位文件”，他就会说“哪有那么方便，我根本不记得文件名、文件内容及其他任何特征，我只能从各种分类文件夹里寻找”，那你就去找呗，你不是我的学生，也不付给我咨询费，也没赞助过MacTalk，我有什么义务让你知道Spotlight的好处呢？



所以关于这个江湖，我最喜欢的两句话送给大家：

自反而缩，虽万千人，吾往矣！
夫唯不争，天下莫能与之争者！

在之前的MacTalk中我介绍过几次Alfred，个人以为小帽子是Mac平台上最为传奇的效率作品，誉为神兵利器毫不为过。其实这个领域当年的带头大哥是Quicksilver，一时风头无两。但是一个人在风头浪尖站太久就会倦怠，而且QS也没找到合适的盈利模式，结果被Alfred迎头赶上，等QS醒过来再发布新版本的时候，江山已经易主，Alfred强势发布2.0，而且通过Powerpack模式的强大功能转化了很多免费用户，目前看来Alfred已经一骑绝尘了。

昨天MacTalk之后有不少读者居然不知道Alfred，所以今天给大家相对系统的介绍一下。

1、安装（不说了去Google吧）。

2、基础快捷键：option+space。

3、打开应用程序：Alfred几乎是一切程序的入口，你再也不需要找妈妈要开始菜单了。用快捷键呼出Alfred，输入任何一款应用程序的中文或英文名称，即可快速定位程序，回车打开。

4、简单查找文件：用快捷键呼出Alfred，键入空格，输入你要查找文件名，即可定位文件，回车打开，command+回车打开文件所在文件夹。

5、复杂操作文件：通过find、open、in等关键词搜索文件。find是定位文件，open是定位并打开文件，in是在文件中进行全文检索，三种检索方式基本上可以找到任何你想找的文件。

6、直接当做计算器使用。

7、操作Shell：输入>即可直接运行shell命令。比如>cat *.py |grep print，可以直接打开终端并查找当前py文件中包含print的语句。

8、输入iTunes，会出现一个iTunes mini play，打开可以通过Alfred控制音乐播放。用快捷键也能完成这个功能：shift+option+command+p。

9、输入email，后面跟邮件地址，可以直接打开写邮件的界面。

10、定义文字片段，在Alfred的设置-Features选中Clipboard，在Snippets里定义自己常用的文字片段，比如代码、地址等等等，之后以option+command+c呼出界面，输入文字片段的关键字回车即可。

11、在option+command+c呼出的界面里还包括剪贴板历史，输入关键字自动匹配。

12、简单搜索：直接输入你要查询的内容，回车即可打开默认浏览器进行搜索。

13、自定义搜索，这个稍微复杂些，打开设置窗口，点击Features-Custom Search，在右侧栏添加自定义搜索。举几个例子帮助大家理解下规则：

（1）搜索iOS App：

Search URL：itunes://ax.search.itunes.apple.com/WebObjects/MZSearch.woa/wa/search?term={query}

Title：iOS App

Keyword：ios

（2）搜索Mac App：

Search URL：macappstore://ax.search.itunes.apple.com/WebObjects/MZSearch.woa/wa/search?q={query}

Title：Mac App

Keyword：mac

设置完之后，呼出Alfred，输入mac dash或ios多看，看看什么效果

（3）翻译：

Search URL：`http://translate.google.cn/#auto/zh-CN/{query}`

Title：英译中

Keyword：en

设置完之后，呼出Alfred，输入en awesome，看看什么效果

大家可以据此自定义各种快捷查询、翻译、打开特定网页等功能。

14、编写自己的插件：Alfred2的推出伴随的是成熟的workflow插件机制，这部分内容就更加复杂一些，这次就不做详细介绍了。我为Alfred贡献了三个workflow，分别是查找本地视频、查找yyets，查找startup news，已经放到了github上，大家可以参考，我之前的MacTalk也介绍过，网址：https://github.com/jackychi

Alfred功能不止于此，以上介绍的功能有的属于收费版有的属于免费版，大家根据自己的需要各取所需吧。

终极Shell

【发布日期　2013年7月24日】

在开始今天的MacTalk之前，先问两个问题吧：
1、相对于其他系统，Mac的主要优势是什么？
2、你们平时用哪种Shell？
……



第一个童靴可以坐下了，Mac的最大优势是GUI和命令行的完美结合，不要把所有注意力放在Mac性感的腰身和明媚的显示屏上好吧，这不是妹纸！第二个童靴你可以出去面壁了，讲了这么多期MacTalk你告诉我还在用Windows的cmd，你让Mac君情何以堪？哪怕你就说在用Linux的Bash我也就原谅你了，踢飞！

上次在“如何学习一门编程语言”里提到了Shell，也有读者问到Shell的问题，所以这次给大家说说Shell的事。

我在“趣谈个人建站”里介绍过一点Shell，自己的东西借用下不丢人，把扯淡的拿掉，干货留下，就是如下内容：

Shell是Linux/Unix的一个外壳，你理解成衣服也行。它负责外界与Linux内核的交互，接收用户或其他应用程序的命令，然后把这些命令转化成内核能理解的语言，传给内核，内核是真正干活的，干完之后再把结果返回用户或应用程序。

Linux/Unix提供了很多种Shell，为毛要这么多Shell？难道用来炒着吃么？那我问你，你同类型的衣服怎么有那么多件？花色，质地还不一样。写程序比买衣服复杂多了，而且程序员往往负责把复杂的事情搞简单，简单的事情搞复杂。牛程序员看到不爽的Shell，就会自己重新写一套，慢慢形成了一些标准，常用的Shell有这么几种，sh、bash、csh等，想知道你的系统有几种Shell，可以通过以下命令查看：

cat /etc/shells

显示如下：

/bin/bash
/bin/csh
/bin/ksh
/bin/sh
/bin/tcsh
/bin/zsh

在Linux里执行这个命令和Mac略有不同，你会发现Mac多了一个zsh，也就是说OS X系统预装了个zsh，这是个神马Shell呢？

目前常用的Linux系统和OS X系统的默认Shell都是bash，但是真正强大的Shell是深藏不露的zsh，这货绝对是马车中的跑车，跑车中的飞行车，史称『终极Shell』，但是由于配置过于复杂，所以初期无人问津，很多人跑过来看看zsh的配置指南，什么都不说转身就走了。直到有一天，国外有个穷极无聊的程序员开发出了一个能够让你快速上手的zsh项目，叫做“oh my zsh”，Github网址是：https://github.com/robbyrussell/oh-my-zsh。这玩意就像“X天叫你学会C++”系列，可以让你神功速成，而且是真的。

好，下面我们看看如何安装、配置和使用zsh。

安装zsh

如果你用Mac，就可以直接看下一节：
如果你用Redhat Linux，执行：sudo yum install zsh
如果你用Ubuntu Linux，执行：sudo apt-get install zsh
如果你用Windows……去洗洗睡吧。

安装完成后设置当前用户使用zsh：`chsh -s /bin/zsh`，根据提示输入当前用户的密码就可以了。

安装oh my zsh

首先安装git，安装方式同上，把zsh换成git即可。
安装“oh my zsh”可以自动安装也可以手动安装。
自动安装：

wget https://github.com/robbyrussell/oh-my-zsh/raw/master/tools/install.sh -O -|sh

手动安装：

git clone git://github.com/robbyrussell/oh-my-zsh.git ~/.oh-my-zsh
cp ~/.oh-my-zsh/templates/zshrc.zsh-template ~/.zshrc

都不复杂，安装完成之后退出当前会话重新打开一个终端窗口，你就可以见到这个彩色的提示了：



配置

zsh的配置主要集中在用户当前目录的.zshrc里，用vim或你喜欢的其他编辑器打开.zshrc，在最下面会发现这么一行字：

`#Customize to your needs…`

可以在此处定义自己的环境变量和别名，当然，oh my zsh在安装时已经自动读取当前的环境变量并进行了设置，你可以继续追加其他环境变量。

接下来进行别名的设置，我自己的部分配置如下：

```
alias cls='clear'
alias ll='ls -l'
alias la='ls -a'
alias vi='vim'
alias javac="javac -J-Dfile.encoding=utf8"
alias grep="grep——color=auto"
alias -s html=mate      #在命令行直接输入后缀为html的文件名，会在TextMate中打开
alias -s rb=mate        #在命令行直接输入ruby文件，会在TextMate中打开
alias -s py=vi          #在命令行直接输入python文件，会用vim中打开，以下类似
alias -s js=vi
alias -s c=vi
alias -s java=vi
alias -s txt=vi
alias -s gz='tar -xzvf'
alias -s tgz='tar -xzvf'
alias -s zip='unzip'
alias -s bz2='tar -xjvf'
```

zsh的牛粪之处在于不仅可以设置通用别名，还能针对文件类型设置对应的打开程序，比如：

alias -s html=mate

意思就是你在命令行输入hello.html，zsh会为你自动打开TextMat并读取hello.html；

alias -s gz='tar -xzvf'

表示自动解压后缀为gz的压缩包。

总之，只有想不到，木有做不到，吓尿了吧。

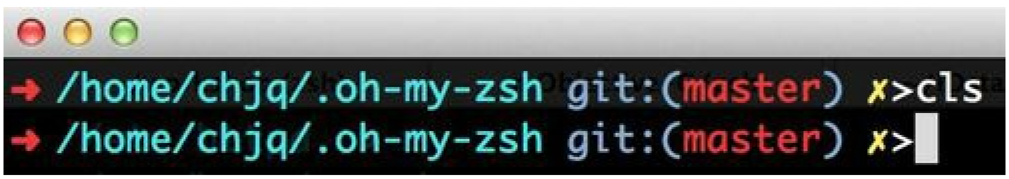
设置完环境变量和别名之后，基本上就可以用了，如果你是个主题控，还可以玩玩zsh的主题。在.zshrc里找到ZSH_THEME，就可以设置主题了，默认主题是：

ZSH_THEME="robbyrussell"

oh my zsh提供了数十种主题，相关文件在~/.oh-my-zsh/themes目录下，你可以随意选择，也可以编辑主题满足自己的变态需求，我采用了默认主题robbyrussell，不过做了一点小小的改动：

PROMPT='％{$fg_bold[red]％}➜％{$fg_bold[green]％}％p％{$fg[cyan]％}％d ％{$fg_bold[blue]％}$(git_prompt_info)％{$fg_bold[blue]％}％％{$reset_color％}>'
#PROMPT='％{$fg_bold[red]％}➜％{$fg_bold[green]％}％p ％{$fg[cyan]％}％c ％{$fg_bold[blue]％}$(git_prompt_info)％{$fg_bold[blue]％}％％{$reset_color％}'

对照原来的版本，我把c改为d，c表示当前目录，d表示绝对路径，另外在末尾增加了一个“>”，改完之后的效果是这样的：



大家可以尝试进行改造，也算个趣事。
最后我们来说说插件。

插件

oh my zsh项目提供了完善的插件体系，相关的文件在~/.oh-my-zsh/plugins目录下，默认提供了100多种，大家可以根据自己的实际学习和工作环境采用，想了解每个插件的功能，只要打开相关目录下的zsh文件看一下就知道了。插件也是在.zshrc里配置，找到plugins关键字，你就可以加载自己的插件了，系统默认加载git，你可以在后面追加内容，如下：

plugins=(git textmate ruby autojump osx mvn gradle)

下面简单介绍几个：
1、git：当你处于一个git受控的目录下时，Shell会明确显示“git”和branch，如上图所示，另外对git很多命令进行了简化，例如gco='git checkout'、gd='git diff'、gst='git status'、g='git’等等，熟练使用可以大大

减少git的命令长度，命令内容可以参考`~/.oh-my-zsh/plugins/git/git.plugin.zsh`

2、textmate：mr可以创建ruby的框架项目，tm finename可以用textmate打开指定文件。

3、osx：tab增强，quick-look filename可以直接预览文件，man-preview grep可以生成grep手册的pdf版本等。

4、autojump：zsh和autojump的组合形成了zsh下最强悍的插件，今天我们主要说说这货。

首先安装autojump，如果你用Mac，可以使用brew安装：

brew install autojump

如果是Linux，去下载autojump的最新版本，比如：

wget https://github.com/downloads/joelthelion/autojump/autojump_v21.1.2.tar.gz

解压缩后进入目录，执行

./install.sh

最后把以下代码加入.zshrc：

[[-s ~/.autojump/etc/profile.d/autojump.sh]]&&. ~/.autojump/etc/profile.d/autojump.sh

至此，安装、配置、插件三位一体，终极Shell全面登场。退出终端会话重新登录，开始感受zsh的迅疾如风！

使用zsh

1、兼容bash，原来使用bash的兄弟切换过来毫无压力，该咋用咋用。

2、强大的历史记录功能，输入grep然后用上下箭头可以翻阅你执行的所有grep命令。

3、智能拼写纠正，输入gtep mactalk *-R，系统会提示：zsh:correct

'gtep'to 'grep'[nyae]？比妹纸贴心吧，她们向来都是让你猜的。

4、各种补全：路径补全、命令补全，命令参数补全，插件内容补全等等。触发补全只需要按一下或两下tab键，补全项可以使用ctrl+n/p/f/b上下左右切换。比如你想杀掉java的进程，只需要输入kill java +tab键，如果只有一个java进程，zsh会自动替换为进程的pid，如果有多个则会出现选择项供你选择。ssh +空格+两个tab键，zsh会列出所有访问过的主机和用户名进行补全。

5、智能跳转，安装了autojump之后，zsh会自动记录你访问过的目录，通过j +目录名可以直接进行目录跳转，而且目录名支持模糊匹配和自动补全，例如你访问过hadoop-1.0.0目录，输入j hado即可正确跳转。j ——stat可以看你的历史路径库。

6、目录浏览和跳转：输入d，即可列出你在这个会话里访问的目录列表，输入列表前的序号，即可直接跳转。

7、在当前目录下输入..或…，或直接输入当前目录名都可以跳转，你甚至不再需要输入cd命令了。

8、通配符搜索：ls -l **/*.sh，可以递归显示当前目录下的shell文件，文件少时可以代替find，文件太多就歇菜了。

9、更强的别名：请参考配置一节。

10、插件支持：请参考插件一节。

……

看完这篇文章，你就知道，zsh一出，无人再与争锋！终极二字不是盖的。

如果你是个正在使用shell程序员，如果你依然准备使用bash，那就去面壁和忏悔吧，别说你订阅过MacTalk！

感谢那位开发了oh my zsh的无聊程序员，他可能没有因此收获物质上的利益，但是他的代码提升了无数程序员的效率，节省了大量的时间，我们说，程序员改变世界！